



(11) **EP 1 845 443 A2**

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
17.10.2007 Bulletin 2007/42

(51) Int Cl.:
G06F 9/44 (2006.01)

(21) Application number: **07007130.3**

(22) Date of filing: **04.04.2007**

(84) Designated Contracting States:
**AT BE BG CH CY CZ DE DK EE ES FI FR GB GR
HU IE IS IT LI LT LU LV MC MT NL PL PT RO SE
SI SK TR**
Designated Extension States:
AL BA HR MK YU

- **Kenntner, Joachim**
69126 Heidelberg (DE)
- **Wagner, Frank**
69234 Dielheim-Horrenberg (DE)
- **Fitzer, Joachim**
69198 Schriesheim (DE)
- **Reiss, Thomas**
69121 Heidelberg (DE)
- **Lang, Klaus-Peter**
74336 Brackenheim-Botenheim (DE)

(30) Priority: **13.04.2006 US 404147**

(71) Applicant: **SAP AG**
69190 Walldorf (DE)

(72) Inventors:
• **Freund, Jens**
69124 Heidelberg (DE)

(74) Representative: **Müller-Boré & Partner**
Patentanwälte
Grafinger Strasse 2
81671 München (DE)

(54) **Software model business process variant types**

(57) Methods and apparatus, including computer program products, to realize a software model are described. Process components are defined that characterize software implementing respective and distinct business processes and additionally define at least one process agent that enables communications between a business object associated with the corresponding proc-

ess component and a business object associated with any other process component. Business Process Variant Types are also defined that associate one or more of the process agents for the corresponding process component so that selection of a process variant type causes the associated one or more process agents to be activated.

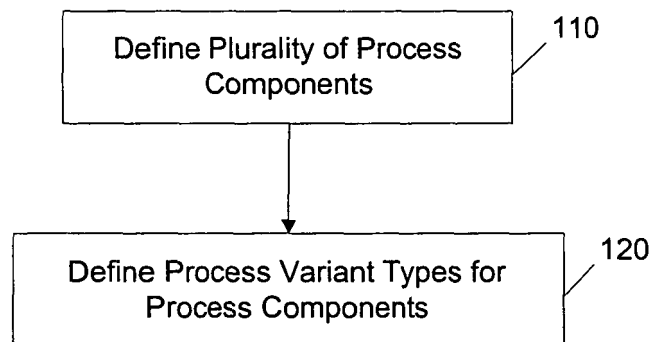


FIG. 1

Description

BACKGROUND

[0001] The subject matter of this patent applications relates to modeling software systems, and more particularly to modeling of Business Process Variant Types in connection with the composition and interaction of components in a software system.

[0002] Enterprise software systems are generally large and complex. Such systems can require many different components, distributed across many different hardware platforms, possibly in several different geographical locations. Typical software modeling systems may not be able to reduce this complexity for end users. In order to design, configure, update or implement an enterprise software system, one is required to understand details of the system at varying levels, depending on his or her role in designing, managing or implementing the system. For example, a systems administrator may need a high-level technical understanding of how various software modules are installed on physical hardware, such as a server device or a network, and how those software modules interact with other software modules in the system. A person responsible for configuring the software may need a high-level functional understanding of the operations that each functional component provides. An application designer may need a low-level technical understanding of the various software interfaces that portions of the application require or implement. And an application developer may need a detailed understanding of the interfaces and functionality he or she is implementing in relation to the remainder of the system.

SUMMARY

[0003] In one aspect, one or more process components can be defined. Each these process components characterize software implementing respective and distinct business processes and can define at least one process agent. Each such process agent enables communications between a business object associated with the corresponding process component and a business object associated with any other process component. One or more Business Process Variant Types can also be defined for at least one for at least one of the process components. Each of the Business Process Variant Types associates one or more of the process agents defined for the corresponding process component so that selection of a process variant type causes the associated one or more process agents to be activated.

[0004] The process components can characterize inbound operations to handle incoming messages associated with a modification of reading of data encapsulated in a business object associated with the process component. One or more of the inbound operations can be a synchronous operation operable to receive a synchronous message generated by an external synchronous

outbound operation defined by an external process component. In some variations, one or more of the inbound operations is operable to receive a message of a first type and convert it into a message of a second type.

[0005] The process components can also characterize outbound operations to handle outgoing messages associated with a modification or reading of data encapsulated in at least one business object associated with another process component. The outbound operations can be called after the business object associated with a corresponding outbound operation is read or modified. One or more of the outbound operations is an asynchronous outbound operation operable to generate an asynchronous message for receipt by an asynchronous inbound operation defined by an external process component. Additionally, in some variations, outbound operations can send messages after they are called.

[0006] The process agents can comprise inbound process agents, outbound process agents, or a combination of both. The inbound process agents can characterize inbound operations to handle incoming messages. The outbound process agents can characterize outbound operations to transmit outgoing messages to an external process component.

[0007] In some variations, a first of the process components is associated with a first deployment unit and a second of the process components is associated with a second deployment unit. Such deployment units can characterize independently operable and deployable software.

[0008] The process components can additionally define service interfaces having pair-wise interactions between pairs of process components. Relatedly, one or more business objects, each which being solely associated with a single process component can be defined. In some implementations, none of the business objects of any one of the process components interacts directly with any of the business objects associated with any of the other process components.

[0009] A plurality of process components can be logically associated to realize a business scenario. This logical association can take the form of an interaction scenario. The logical association is dependent on the selected Business Process Variant Types as process components only interact, in some variations, based on the activated outbound process agents. As a result, interactions between process components not having a process variant type activating process agents coupled the process components can be limited. Business Process Variant Types can also be used to verify that connected process components have corresponding Business Process Variant Types activating relevant outbound process agents.

[0010] In an interrelated aspect, a plurality of process agents can be defined for each of two process components. Each process agent is either an inbound process agent or an outbound process agent. An inbound process agent is operable to receive a message from an inbound operation. An outbound process agent is operable to

cause an outbound operation to send a message. Thereafter, interactions between at least one inbound process agent of a first process component and at least one outbound process agent of a second process component are defined. Additionally, interactions between at least one inbound process agent of the second process component and at least one outbound process agent of the first process component are defined. One or more Business Process Variant Types are defined for at least one of the process components. Each of the Business Process Variant Types associates one or more of the process agents defined for the corresponding process component. Selection of a process variant type causes the associated one or more process agents to be activated.

[0011] In a further interrelated aspect, a process interaction map is displayed in a first view, a process component architectural design is illustrated in a second view, and a process component interaction architectural design is illustrated in a third view. The process interaction maps illustrates interactions among a plurality of process components linked together by selected Business Process Variant Types. Each of the process components characterizes software implementing a respective and distinct business process, and each of the process components defines a respective service interface for interacting with other process component. The process component architectural design illustrates an inbound part, a business object part, and an outbound part for a selected process component. The inbound part identifies all external process components that use one or more inbound operations of the selected process component. The business object part identifies all business objects associated with the selected process component. The outbound part identifies all external process components utilized by one or more outbound operations of the selected component. The process component interaction architectural design illustrates message transfer between exactly two process components to modify or read business objects associated with each business object. Each process component illustrates a plurality of process agents, each process agent being either an inbound process agent or an outbound process agent. The inbound process agent is operable to receive a message from an inbound operation. The outbound process agent is operable to cause an outbound operation to send a message. In addition, only outbound process agents associated with a selected process variant type are activated.

[0012] Computer program products, which can be tangibly encoded on computer readable-material, are also described. Such computer program products can include executable instructions that cause a computer system to implement one or more of the acts and/or components described herein.

[0013] Similarly, computer systems are also described that can include a processor and a memory coupled to the processor. The memory can encode one or more programs that cause the processor to implement one or more of the acts and/or components described herein.

[0014] The subject matter described herein provides many advantages. A model provides modeling entities to represent aspects of a software system. Multiple views of the model are provided in a user interface. The model views offer varying levels of detail, allowing users to focus on the information that is important for their task. Model entities can be reused and correspond to reusable software that implements functionality corresponding to the model entity. The model supports dynamic mapping between incompatible message formats. A model can incorporate external components. The models can be used to generate metadata, which can be stored in a repository and used in various downstream processes and tools.

[0015] Moreover, the subject matter described herein provides a logical abstraction of how various software modules can interact to effect a business scenario. In particular, effective use can be made of process components as units of software reuse, to provide a design that can be implemented reliably in a cost effective way. Deployment units, each of which is deployable on a separate computer hardware platform independent of every other deployment unit, enable a scalable design. Furthermore, service interfaces of the process components can define a pair-wise interaction between pairs of process components that are in different deployment units in a scalable manner.

[0016] One implementation of the subject matter described in this specification provides all of the above advantages.

[0017] Details of one or more implementations of the subject matter described in this specification are set forth in the accompanying drawings and in the description below. Further features, aspects, and advantages of the subject matter will become apparent from the description, the drawings, and the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0018]

FIG. 1 is an illustration of a modeling method;
 FIG. 2 is an illustration of a modeling system;
 FIG. 3 is an illustration of process component modeling entities.
 FIGS. 4A-4C are illustrations of a process interaction map;
 FIG. 5 is an illustration of a process component model;
 FIG. 6 is an illustration of a process component interaction model;
 FIGS. 7A-7B are illustrations of a business object map;
 FIG. 8 is an illustration of an integration scenario model entity;
 FIGS. 9-9A are illustrations of an integration scenario catalog;
 FIGS. 10A-10B are illustrations of a GUI for presenting one or more graphical depictions of views of a

model and modeling entities;

FIG. 11 is an illustration of process component interaction with an external process component;

FIG. 12 is an illustration of process component interaction through a mapping model element;

FIG. 13 is an illustration of a partial integration scenario based on a cash sales invoice process variant type;

FIG. 14 is an illustration of a partial integration scenario based on a standard customer invoice process variant type;

FIG. 15 illustrates Business Process Variant Types and outbound process agents for a customer invoice processing process component;

FIG. 16 illustrates sample process components and their respective Business Process Variant Types; and

FIG. 17 illustrates an integration scenario in which selected Business Process Variant Types are displayed within each process component.

[0019] Like reference numbers and designations in the various drawings indicate like elements.

DETAILED DESCRIPTION

[0020] In the context of this document, a model is a representation of a software system, part of a software system, or an aspect of a software system. A model can be associated with one or more views. A view of a model represents a subset of the information in the model. For purposes of discussion, the term "model" will be used to refer to both a model or a view of the model. A modeling system can be used to create, modify and examine a model. A model can be used in a software development process to describe or specify a software application, or parts or aspects of a software application, for developers implementing or modifying the application. The model specifies the design to a useful level of detail or granularity. A compliant implementation of the modeled functionality will conform to the specification represented by the model.

[0021] FIG. 1 is a process flow diagram illustrating a method 100, at which, at 100, one or more process components are defined. Each of the process components characterizes software implementing respective and distinct business processes and defines at least one process agent. Each process agent enables communications between a business object associated with the corresponding process component and a business object associated with any other process component. Thereafter, at 120, one or more Business Process Variant Types for at least one of the process components is defined. Each of the Business Process Variant Types associates one or more of the process agents defined for the corresponding process component so that selection of a process variant type causes the associated one or more process agents to be activated.

[0022] FIG. 2 illustrates a modeling system 200. An interactive graphical user interface (GUI) 204 allows a user to create, inspect and modify a model. The GUI 204 can present a model in different views offering differing levels of detail. This arrangement allows users to focus on information that is appropriate to their role or the task at hand. A model design component 206 coupled to the GUI 204 provides one or more tools for modifying and manipulating a model, as will be discussed below. A repository 202 is capable of storing one or more models and associated information. By way of illustration and without limitation, the repository can incorporate one or more files, databases, services, combinations of these, or other suitable means for providing persistent storage of model information..

[0023] FIG. 3 is an illustration of process component modeling entities (or "process components") in a model. For brevity, where the sense is clear from the context, the term "process component" will be used to refer both to the modeling entity and to an implementation in a software system of a process represented by that modeling entity. The same dual use will be made of other terms to refer both to the modeling entity and an implementation represented by the entity, where the meaning is clear from the context.

[0024] A process component is a software package that realizes a business process and exposes its functionality as services. The functionality contains business transactions. A process component contains one or more semantically related business objects (e.g., 330, 310). A business object belongs to no more than one process component.

[0025] Process components are modular and context-independent. Context-independent means that a process component is not specific to a given integration scenario (integration scenarios are described later.) Therefore, process components are reusable, that is, they can be used in different integration scenarios.

[0026] A process component has one or more service interface modeling entities (316, 318, 320, 322, 336, 338) (or "interfaces"). An interface is a named grouping of one or more operations. It specifies offered (inbound service interface) or used (outbound service interface) functionality. While in general process components will have service interfaces, it is permissible to define a process component having no service operations. This would be appropriate, for example, for process components that inherently or by design interact only with process components deployed on the same hardware platform, in which circumstances a non-service method of interacting, e.g., through shared memory or database records, might be preferred.

[0027] An operation belongs to exactly one process component. A process component generally has multiple operations. An operation is the smallest, separately-callable function, described by a set of data types used as input, output, and fault parameters serving as a signature. An operation can use multiple message types for

inbound, outbound, or error messages. An operation is specific to one interface, i.e., the same operation cannot be used in more than one interface.

[0028] Operations are described for purposes of exposition in terms of process agents. A process agent (or "agent") is an optional modeling entity representing software that implements an operation. Operations can be implemented through other conventional techniques. Operations (and hence agents) can be synchronous or asynchronous, and inbound or outbound. As will be described below, a process component can characterize more process agents than are required for a particular implementation.

[0029] Synchronous outbound operations send synchronous request messages and process response messages. Synchronous inbound operations respond to messages from synchronous outbound operations. Synchronous communication is when a message is sent with the expectation that a response will be received promptly. Asynchronous communication comes with the expectation that a response will be provided by a separate operation invoked at a later point in time.

[0030] An asynchronous outbound operation is specific to a sending business object. If the asynchronous outbound operation is triggering a new communication to another process component, it is specific for the triggered process component. However, the same asynchronous outbound process operation can be used for two operations which are part of the same message choreography. If the asynchronous outbound operation is sending only a confirmation (not triggering), it might be re-used for different receiving process components.

[0031] Inbound operations are called after a message has been received. Based on a business object's status, inbound operations may initiate communication across deployment units, may initiate business-to-business (B2B) communication, or both by sending messages using well-defined services.

[0032] The model can describe the potential invocation by one process component of an operation on another process component. Graphically, this is depicted as an arc (340, 342) in FIG. 3 connecting the two process components 306 and 308. Invocation of an operation on a process component is always accomplished by another process component sending a message to the process component, if the two process components are part of different deployment units, which are described below. Interaction between two process components in the same deployment unit, on the other hand, can be implemented by the passing of messages, as described, or it can be implemented by the use of resources, e.g., data objects, database records, or memory, that are accessible to both process components when they are deployed.

[0033] Messages are described by message modeling entities (or "messages") in the model.

[0034] A process agent can be associated with a single interface. For example, interface 338 is associated with process agent 332, interface 336 is associated with proc-

ess agent 334, interface 316 is associated with process agent 312, and interface 318 is associated with process agent 314. In one variation, each operation is associated with a process agent.

[0035] An output operation generally responds to an action (e.g., create, read, update, delete, etc.) with a business object associated with the operation. The operation will generally perform some processing of the data of the business object instance whose change triggered the event. An outbound operation triggers subsequent business process steps by sending messages using well-defined outbound services to another process component, which generally will be in another deployment unit, or to a business partner. For example, outbound process agent 324 in process component 306 can invoke an operation of interface 322 to send a message that will be received by the inbound process agent 312 in process component 308. The message is routed to a specific operation in interface 316 according to the signature or type of the message, which the inbound process agent 312 handles.

[0036] Inbound process agents when implemented are pieces of software that are used for the inbound part of a message-based communication. An inbound process agent starts the execution of the business process step requested in a message by creating or updating one or multiple business object instances, e.g., for associated business objects (330, 310) in response to receiving a message. Outbound process agents when implemented can send messages in response to a business object changing or interaction with a business object. For example, the inbound process agent 312 may modify business object 310, thus triggering outbound process agent 314 to send a message to the inbound process agent 328. If two operation invocations are part of the same message choreography, they are associated with the same process agent.

[0037] A business object model entity models a business object. A business object is a representation of a type of a uniquely identifiable business entity (an object instance) described by a structural model and zero or more service interfaces. Implemented business processes operate on business objects.

[0038] A business object represents a specific view on some well-defined business content. A business object represents content, which a typical business user would expect and understand with little explanation. Business objects are further categorized as business process objects and master data objects. A master data object is an object that encapsulates master data (i.e., data that is valid for a period of time). A business process object, which is the kind of business object generally found in a process component, is an object that encapsulates transactional data (i.e., data that is valid for a point in time). The term business object will be used generically to refer to a business process object and a master data object, unless the context requires otherwise. Properly implemented, business objects are implemented free of re-

dundancies.

[0039] Business process objects are associated with exactly one process component. Master data objects are either associated with exactly one process component or exactly one deployment unit.

[0040] Business objects residing in a foundation layer are called business foundation objects. The foundation layer is deployed on every platform, and its business objects, process components, and reuse services are available to be used by all application scenarios. It is assumed that business objects in the foundation layer will be local in all integration scenarios and can be directly accessed synchronously from business objects within deployment units in an application layer. Business objects in the foundation layer can be associated with more than one process component. Process components in the foundation layer have no BPVTs as such process components do not have process agents, and messages - only direct calls.

[0041] FIGs. 4A-4C are illustrations of a process interaction map 400. A process interaction map is a modeling entity that describes interactions between two or more process components. It can be presented in the GUI 204 (FIG. 2) by the model design component 206 as a circuit diagram, for example, with arcs indicating potential interactions between process components. In a visual rendition of the map 400, process components are represented as icons (e.g., 404, 406, 408). So called "external" process components are indicated with dashed lines (e.g., 406, 408). External process components are shown to place the modeled process components in their operational context relative to another system, e.g., a system belonging to another company, such as a customer or other third party. The GUI 204 allows a user to connect and disconnect process components (i.e., to indicate potential interactions), move process components, and zoom in a specific portion of the map 400 to see more detail, as indicated by view 414.

[0042] Groups of process components can be organized into scenarios and deployment units. An integration scenario modeling entity (or "scenario") describes a group of process components that interact directly or indirectly (i.e., through one or more other process components) with each other. A process component belongs to one deployment unit. Scenarios are discussed below.

[0043] A deployment unit modeling entity (e.g., 402, 410, 412) models a deployment unit, which includes one or more process components that can be deployed together on a single computer system platform.

[0044] Separate deployment units can be deployed on separate physical computing systems and include one or more process components. For example, a physical system can be a cluster of computers having direct access to a common database. The process components of one deployment unit interact with those of another deployment unit only using messages passed through one or more data communication networks or other suitable communication channels. Thus, a deployment unit soft-

ware entity deployed on a platform belonging to *Company A* can interact with a deployment unit software entity deployed on a separate platform belonging to *Company B*, allowing for business-to-business communication. Or deployment units in different divisions of the same company can interact with each other. More than one instance of a given deployment unit software entity can execute at the same time.

[0045] FIG. 5 is an illustration of a process component model (PCM) 500. A PCM is a view of a model that incorporates the model entities associated with a particular process component. A PCM can also describe potential interactions between a process component and other process components in the same or in different deployment units. For example, the illustrated process component 500 can interact with a Customer Requirement Processing component 504 and a Customer Invoice Processing component 525. Moreover, a PCM can describe interaction with external process components that are controlled by third parties (e.g., 528).

[0046] The PCM models operations incorporated in a process component. For example, inbound operation Change Sales Order based on Customer Requirement Fulfillment Confirmation 508, and outbound operations Request Invoicing 520 and Confirm Sales Order 522. The arc 530 connecting the process component 504 to the interface 502 represents that the process component 504 can invoke an operation on that interface. The arcs 532 and 534 represent that the process component illustrated in 500 can invoke an operation on process components 525 and 528, respectively.

[0047] The PCM optionally models process agents (e.g., 510, 516, 518) corresponding to the process component's operations. For example, the Change Sales Order based on Customer Requirement inbound process agent 510 models processing or responding to a message routed to inbound operations 508 or 540. The inbound process agent 510, for example, will access and modify the Sales Order business object 514 as part of the processing, e.g., change the delivery date of goods or services on the sales order.

[0048] Process component 525 can receive one or more messages by way of outbound operation 520, as denoted by the arc 532 connecting outbound operation 520 to the process component 525. Based on the change associated with the business object 514, the Request Invoicing from Sales Order to Customer Invoice Processing outbound process agent 518 invokes operation 520 in interface 526 to send a message to process component 525. Likewise, external process component 528 can receive one or more messages sent by outbound operation 522, as denoted by the arc 534 connecting operation 522 to the process component 528. Based on the state or a state change associated with the business object 514, outbound process agent 516 can invoke operation 522 to send a message to external process component 528.

[0049] FIG. 6 is an illustration of a process component interaction model (PCIM) 600. PCIMs can be reused in

different integration scenarios. A PCIM is a view of a model that incorporates relevant model entities associated with potential interaction between two process components (e.g., 602, 604). Interfaces, process agents and business objects that are not relevant to the potential interaction are excluded. The PCIM 600 shows interactions between a Time and Labor Management process component 602 and a Goods and Service Acknowledgement process component 604.

[0050] The Time and Labor Management process component 602 includes an Employee Time Calendar business object 606 that gives a read-only information of a calendar based overview of different time data (e.g., Planned working time, an absences and working time confirmation) of employees and their superposition (e.g., illness, vacation, etc). The Employee Time Calendar business object 606 may use a Notify Goods and Services Acknowledgement outbound process agent 608 to invoke a Notify of Goods and service Acknowledgement Notification operation 610 or a Notify of Goods and Service Acknowledgement Cancellation operation 612, which are both included in the Internal Service Acknowledgement Out interface 614. The Notify of Goods and service Acknowledgement Notification operation 610 notifies the Goods and Service Acknowledgement process component 604 of a service provided by an external employee. The Notify of Goods and service Acknowledgement Notification operation 610 sends a Goods and Service Acknowledgement Request message 616 when an active employee time with Goods and Service Acknowledgement relevant information is created or changed.

[0051] The Goods and Service Acknowledgement process component 604 receives the Goods and Service Acknowledgement Request message 616 via an Internal Acknowledgement In interface 618. Upon receipt of the Goods and Service Acknowledgement Request message 616, a Create Goods and Service Acknowledgement operation 620 is invoked to create Goods and service Acknowledgement, and Time and Labor Management by initiating a Maintain GSA based on Internal Acknowledgment inbound process agent 622. The Maintain GSA based on Internal Acknowledgment inbound process agent 622 updates or creates a Goods and Service Acknowledgement business object 624 to report the receipt of goods and services. The Goods and Service Acknowledgement business object 624 may be used when employees of a company can confirm that they have received the goods and services they ordered through internal requests, purchasers, or designated recipients of goods and services, can confirm that they have received the goods and services they ordered on behalf of the employees for whom they are responsible, or suppliers or service providers can report that they have delivered the requested goods, or have rendered they requested services.

[0052] The Notify Goods and Services Acknowledgement outbound process agent 608 may also invoke the Notify of Goods and Service Acknowledgement Cancel-

lation operation 612 to notify the Goods and Service Acknowledgement process component 604 of a cancellation of goods and service. The Notify of Goods and Service Acknowledgement Cancellation operation 612 sends a Goods and Service Acknowledgement Cancellation Request message 626 when an active employee time with Goods and Service Acknowledgement relevant information is cancelled. Upon receipt of the Goods and Service Acknowledgement Cancellation Request message 626, a Cancel Goods and Service Acknowledgement operation 628 is invoked to cancel Goods and service Acknowledgement. Next, the Maintain GSA based on Internal Acknowledgment inbound process agent 622 updates the Goods and Service Acknowledgement business object 624 to report the cancellation of goods and services.

[0053] The message format of a message sent by an outbound operation need not match the message format expected by an inbound operation. If the message formats do not match, and the message is transformed, or mapped. Message mapping is indicated by interposition of an intermediary mapping model element between the source and the destination of the message in a PCM or a PCIM (see below).

[0054] FIGs. 7A-7B are illustrations of a business object map 700. A business object map is a view of a model that incorporates deployment units, process components, and business objects. Interfaces, operations and process agents are excluded from the view. Each model entity is only represented once in the business object map. Hence, the business object map is a representation of all deployment units, process components, and business objects. In the illustrated business object map 700, and as shown in the highlighted portion 728 illustrated in FIG. 7B, a Customer Invoice Processing process component 726 in Customer Invoicing deployment unit 704 incorporates two business objects: a customer invoice request 710 and a customer invoice 708. A Project Processing process component 724 in a Project Management deployment unit 706 includes five business objects: a Project Request 718, a Project 720, a Project Snapshot 712, a Project Simulation 714, and a Project Template 722.

[0055] FIG. 8 is an illustration of an integration scenario model entity 800 (or "integration scenario"). An integration scenario is a realization of a given end-to-end business scenario. It consists of the process components and the interactions between them, which are required for its realization. A process component is only represented once in an integration scenario model, even though the actual flow in the software system might invoke the same process component multiple times. An integration scenario model entity describes at a high level the potential interaction between process components in one or more deployment units that are relevant to realization of the business scenario. For example, an integration scenario can be a set of process components and their interactions working together to realize a business scenario to

achieve a business objective, such as selling products to generate revenue. Internal details of process components are not described, nor are details of process component interactions (e.g., interfaces, operations and messages).

[0056] The illustrated integration scenario 800 is for a service procurement software application. The service procurement application is software that implements an end-to-end process used to procure services. The scenario 800 includes nine deployment units: a Financial Accounting unit 802, a Project Management unit 804, a Purchasing unit 806, a Supplier Invoicing unit 808, a Payment unit 810, a RFQ Processing unit 812, a Due Item Management unit 814, a Requisitioning unit 816, and a Human Capital Management unit 818.

[0057] The Financial Accounting deployment unit 802 includes an Accounting process component 803 that records all relevant business transactions.

[0058] The Project Management deployment unit 804 includes a Project Processing component 820 that is responsible for structuring, planning, and executing measures or projects (e.g., short-term measures, complex projects, etc).

[0059] The Purchasing deployment unit 806 includes four process components: a Purchase Request Processing process component 828, a Purchase Order Processing process component 830, a Purchasing Contract process component 832, and a Goods and Service Acknowledgement process component 833.

[0060] The Purchase Request Processing process component 828 provides a request or instruction to the purchasing department to purchase specified goods or services in specified quantities within a specified time.

[0061] The Purchase Order Processing process component 830 includes a purchase order business object and a purchase order confirmation business object. The purchase order is a request from a purchaser to an external supplier to deliver a specified quantity of goods, or perform a specified service within a specified time. The purchase order confirmation is a communication from a supplier to a purchaser to advise that a purchase order has been received. In particular, a purchase order confirmation may advise the purchaser of the supplier accepting the purchase order, or the supplier proposing changes to the purchase order, or the supplier not accepting the purchase order.

[0062] The Purchasing Contract process component 832 handles an agreement between a purchaser and a supplier that details the supply of goods or the performance of services at agreed conditions. The Purchasing Contract process component includes the purchasing contract business object.

[0063] The Goods and Service Acknowledgement 833 includes a Goods and Service Acknowledgement business object. The Goods and service Acknowledgement business object is a document that states the recipient's, for example, a purchaser's, obligation to pay the supplier for goods received or services rendered. An invoice is

normally created after the goods and service acknowledgement has been confirmed.

[0064] The Supplier Invoicing deployment unit 808 includes a Supplier Invoice Processing process component 836. The Supplier Invoice Processing process component 836 includes a supplier invoice business object and a supplier invoice request business object. The supplier invoice is a document that states the recipient's obligation to pay the supplier for goods received or services rendered. The invoice may be created after the goods and service acknowledgment has been confirmed. The supplier invoice request is a document that is sent to invoice verification, advising that an invoice for specified quantities and prices is expected and may be created through evaluation settlement. The system uses the invoice request as a basis for invoice verification, as well as for the automatic creation of the invoice. The Payment deployment unit 810 includes a Payment Process component 838. The Payment Processing process component 838 is used to handle all incoming and outgoing payments as well as represent the main database for a liquidity status.

[0065] The RFQ deployment unit 812 includes an RFQ Processing process component 840. An RFQ Processing deployment unit includes a Request for Response business object and a quote business object. The request for quotation (RFQ) is a description of materials and services that purchasers use to request responses from potential suppliers. Requests for Quotation can be one of the following types: a request for (price) information, a request for quote that may run over a certain period of time, a request for proposal in complex purchasing situation or live auctions that may be performed over a short time frame. The quote is a response to a request for quotation in which a supplier offers to sell goods and services at a certain price. The quote can be subject to complex pricing and conditions.

[0066] The Due Item Management deployment unit 814 includes a Due Item Processing process component 842. The Due Item Processing process component 842 is used to manage all payables, receivables from service and supply and corresponding sales including a withholding tax.

[0067] The Requisitioning deployment unit 816 includes an Internal Request Processing process component 844. The Internal Request Processing deployment unit 816 includes an Internal Request business object. Employees of a company may make an internal request for the procurement of goods or services for the company. For example, the employees may order stationary, computer hardware, or removal services by creating an internal request. The internal request can be fulfilled by an issue of a purchase request to the purchasing department, a reservation of goods from stock, or a production request.

[0068] The Human Capital Management deployment unit 818 includes a Time and Labor Management process component 848. The Time and Labor Management proc-

ess component 848 supports the definition of employees' planned working time as well as the recording or the actual working times and absences and their evaluation.

[0069] The foundation layer includes a Source of Supply Determination process component 834, a Customer Invoice Processing at Supplier process component 837, a Sales Order Processing at Supplier process component 846, a Payment Processing at Business Partner process component 850, a Bank statement create at bank process component 852, and a Payment order processing at house bank process component 854.

[0070] The service procurement design includes a Source of Supply Determination process component 834 that uses two business objects to determine a source of supply: a supply quota arrangement business object, and a source of supply business object. A supply quota arrangement is a distribution of material requirements or goods to different sources of supply, business partners, or organizational units within a company. An example of the use of supply quota arrangements is the distribution of material requirements between in-house production and different sources for external procurement. A supply quota arrangement can also define the distribution of goods to customers in case of excess production or shortages. A source of supply is an object that describes a logical link between a possible source of products and a possible target.

[0071] A number of external process components, described below, will be used to describe the architectural design. These include a Customer Invoice Processing at Supplier process component 837, a Sales Order Processing at Supplier process component 846, a Payment Processing at Business Partner process component 850, a Bank statement create at bank process component 852, and a Payment order processing at house bank process component 854.

[0072] The Supplier Invoicing deployment unit 808 receives messages from a Customer Invoice at Supplier processing component 837, which is used, at a supplier, to charge a customer for the delivery of goods or services.

[0073] The service procurement design includes a Sales Order Processing at Supplier process component 846 that may receive messages from the RFQ Processing process component 840. The Sales Order Processing at Supplier process component 846 handles customers' requests to a company for delivery of goods or services at a certain time. The requests are received by a sales area, which is then responsible for fulfilling the contract.

[0074] The Payment Processing at Business Partner process component 850, the Bank statement create at bank process component 852, and the Payment order processing at house bank process component 854 may interact with the Payment Processing process component 838. The Payment Processing Process component 838 may send updates to a Payment Processing at Business Partner processing component 850, which is used to handle, at business partner, all incoming and outgoing payments and represent the main data base for the li-

quidity status. The Payment Processing Process component 838 also receives messages from the Bank statement creates at bank process component 852. The message may include a bank Statement for a bank account. The Payment Processing Process component 838 send messages to the Payment order processing at house bank process component 854. The message may include a Bank Payment Order that is a Payment Order which will be sent to a house bank. The bank payment order may contain bank transfers as well direct debits.

[0075] The connector 829 symbol is a graphical convention to improve graphical layout for human reading. A connector is a placeholder for another process component. For example, the connector 829 could be a placeholder for an Accounting process component.

[0076] FIGs. 9A-9B are illustrations of an integration scenario catalog (or "scenario catalog") 900. A scenario catalog presents an organized view of a collection of integration scenarios. The view can be organized in a number of ways, including hierarchically or associatively based on one or more attributes of the integration scenarios. The illustrated integration scenario catalog 900 represents a structured directory of integration scenarios. For example, a scenario directory Sell from Stock 902 representing a family of scenarios includes two entries: a reference to a Sell from Stock integration scenario 904, and a reference to a Sell from Stock for Delivery Schedules integration scenario 906.

[0077] FIG. 10 is an illustration of the GUI 204 (from FIG. 2) for presenting one or more graphical depictions of views of a model and modeling entities. Each view can present a different level of detail or emphasize a different aspect of the model. This allows for different classes of users to focus on the information that is important for carrying out their duties without being distracted by extraneous detail. One or more of the following graphical depictions can be presented: a scenario catalog 1002, an integration scenario model 1004, a PCIM 1008, and a PCM 1010. In one variation, the GUI 204 allows a user to "drill down" to increasing levels of model detail. For example, selection of a scenario icon 1006 in the integration scenario catalog 1002 can cause an associated integration scenario model 1004 to be presented. Selection of a graphical representation of a process component 1014 in the integration scenario can cause an associated PCM 1010 for the process component to be presented. Likewise, selection of an arc 1012 connecting process components in different deployment units can cause a PCIM 1008 for the process components connected by the arc to be presented.

[0078] In one implementation, the aforementioned graphical depictions can be presented singularly or in combination with each other in the GUI 204. Moreover, a given graphical depiction can present all of its underlying information or a portion thereof, while allowing other portions to be viewed through a navigation mechanism, e.g., user selection of a graphical element, issuance of a command, or other suitable means.

[0079] Information can also be represented by colors in the display of model entities. For example, color can be used to distinguish types of business objects, types of process agents and types of interfaces.

[0080] FIG. 11 is an illustration of process component interaction with an external process component, representing an external system. As discussed earlier, a process component can interact with an external process component. This interaction can be modeled even though the interfaces of the external process are unknown, as is the case in this example. (However, if this information is known, it can be incorporated into the model.)

[0081] In this example, potential interactions are shown between a Purchase Order Processing process component 1102 and an external Sales Order Processing at Supplier process component 1104. The Purchase Order Processing process component 1102 includes a Purchase Order business object 1106 which is a request from a purchaser to an external supplier to deliver a specified quantity of goods, or perform a specified service, within a specified time. The Request Purchase Order to Supplier outbound process agent 1108 can request invocation of a Request Purchase Order Creation operation 1112, a Request Purchase Order Cancellation operation 1114, or a Request Purchase Order Change operation 1116 in an Ordering Out interface 1110.

[0082] The Request Purchase Order Cancellation operation 1114 requests a Cancellation of a Purchase Order that was formerly ordered at a supplier which creates a Purchase Order Cancellation Request message 1118. The Request Purchase Order Change operation 1116 requests a change of a purchase order that was formerly ordered at the supplier which creates a Purchase Order Change Request message 1120. The Request Purchase Order Creation operation 1112 requests a Purchase Order from a Supplier which creates a Purchase Order Change Request 1122.

[0083] Upon receiving a create, a change, or a cancellation message, the Sales Order Processing process component 1104 may create a Purchase Order Confirmation message 1123 to update the Purchase Order Processing component 1102. To complete the update, a Create Purchase Order Confirmation operation 1124, included in an Order In interface 1125, may transfer the update to the Purchase Order Confirmation business object 1128 by using a Create Purchase Order inbound process agent 1126. The Purchase Order Confirmation business object 1128 is a confirmation from an external supplier to the request of a purchaser to deliver a specified quantity of material, or perform a specified service, at a specified price within a specified time.

[0084] FIG. 12 is an illustration 1200 of process component interaction through a mapping model element 1214 (or "mapper"). As discussed above, if message formats between two process component operations do not match, the message can be transformed by a mapper on its way from the outbound process agent to the inbound

process agent. For example, output process agent 1216 associated with process component 1202 can send a message 1210 to inbound process agent 1217 in process component 1204 by way of operation 1218 in interface 1206. If the message format associated with operation 1218 does not match that of operation 1220, a transformation of the message from its original format to a format compatible with operation 1220 can be described by a mapper 1214 interposed between the two process agents. The mapper 1214 generates a new message 1212 based on the original message 1210, where the new message has a format that is compatible with operation 1220.

[0085] Depending on the integration scenario being implemented, different message choreographies among process components may be required. However, only specific outbound process agents need to be activated to enable these message choreographies. Often, there can be numerous outbound process agents which are defined by a process component which are not needed for a particular integration scenario. Leaving outbound process agents active even though they are not being utilized has some drawbacks. For example, although a relevance condition of an outbound process agent is coded so that it can be determined that the process component is not active, calling all such outbound process agents and evaluating all of the relevance conditions can degrade performance. Moreover, coding such relevance conditions increases development efforts and system complexity. In another example, if a relevance condition is not defined or is not checked, sending messages to an inactive (not configured) process component will produce a large number of errors.

[0086] In some variations, process component can have one or more associated business Business Process Variant Types ("BPVTs"). A BPVT is a modeling entity that represents a typical way of processing within a process component from a business point of view. A BPVT defines which process agents of a process component are activated, which in turn defines other process components with which the process component interacts. The criteria to define a BPVT can include associated message choreographies or business configuration parameters belonging to different BPVTs of a process component. In some cases, BPVTs of one process component can have the same message choreography as another process component while the business configuration settings differ. As a result, the message choreography of a process component is different for different BPVT of such a process component. There are two categories of BPVTs: main BPVTs which are unique for a particular business object or a business object node and additional BPVTs which are optional and only exist in combination with a main BPVT.

[0087] During application run-time the BPVT will be derived (based on business configuration settings, master data, incoming messages or user interaction) and stored in a business object node. The derived BPVT

could be passed in a message or used as one of the parameters in a relevance condition of outbound process agent. BPVTs can be passed in connection with service and support error tickets (to allow, for example, an embedded support scenario).

[0088] BPVTs can be used to ensure consistency among different deployments based on the modeling methodologies described herein. Moreover, BPVTs can be used to verify that process interaction models associated with a particular integration scenario have been correctly assigned.

[0089] FIGs. 13 and 14 respectively illustrate partial integration scenarios 1300, 1400 in which a customer invoice processing process component 1302 includes different selected BPVTs. In a first variation illustrated in FIG. 13, a cash sales invoice BPVT 1304 is selected. With cash sales invoice the customer invoice processing 1302 interacts with a due item processing process component 1306, a confirmation and inventory process component 1308, a payment processing process component 1310, and an accounting process component 1312 via four outbound process agents. However, in a second variation illustrated in FIG. 14, a standard customer invoice BPVT 1402 is selected. With the standard customer invoice BPVT 1402, the customer invoice processing process component 1302 interacts only with the due item processing process component 1306 and the accounting process component 1312 via two outbound process agents.

[0090] The relationship between the two variations illustrated in FIGs. 13 and 14 is further depicted in FIG. 15. FIG. 15 provides a sample graphical user interface in which a BPVT is related to one or more outbound process agents. With this implementation, the customer invoice processing process component 1302 is illustrated as including both the standard customer invoice BPVT 1402 which has two associated outbound process agents and the cash sales invoice 1304 which has four outbound process agents. In this implementation, the standard customer invoice BPVT 1402 is selected so that connections are provided in a process agent framework activating a notify customer invoice to accounting outbound process agent 1504, and a notify customer invoice to due item processing outbound process agent 1506 via a process agent framework 1502. However, a notify customer invoice to payment outbound process agent 1508 and a notify customer invoice to inventory outbound process agent 1510 are not activated as such outbound process agents 1508, 1510 are only utilized by the cash sales invoice BPVT 1304.

[0091] FIG. 16 is an illustration of a process component catalog 1600. The process component catalog 1600 presents an organized view of a collection of process components 1602 and their associated BPVTs 1604. The view can be organized in a number of ways, including hierarchically or associatively based on one or more attributes of the process components. Selection of a BPVT 1602 will cause only those outbound process agents as-

sociated with the BPVT 1604 to be activated, which in turn results, in defining associated process component interaction models, process agents, and connectivity configurations.

[0092] FIG. 17 illustrates a sample scenario integration model 1700 that includes a plurality of interconnected process components (with the arrows defining outbound operations having corresponding outbound process agents). However, it will be appreciated that many variations of the integration model are available and that some of the identified process components and associated BPVTs may be omitted and other process components and BPVTs may be added depending on the desired implementation. The integration model 1700 includes a supply chain control deployment unit 1702, a customer relationship management deployment unit 1704, a logistics execution deployment unit 1706, a customer invoicing deployment unit 1708, a due item management deployment unit 1710, a financial accounting deployment unit 1712, and a payment deployment unit 1714. The supply chain control deployment unit 1702 comprises a customer requirement processing process component 1716 that is limited by a standard customer requirement BPVT 1718, a supply and demand matching deployment unit 1720 that is limited by a consumption based planning BPVT 1722, and a logistics execution control process component 1724 that is limited by a outbound delivery trigger and response BPVT 1726.

[0093] The customer relationship management deployment unit 1704 includes a customer quote processing (which is an optional process component in integration scenario sell-from-stock") deployment unit 1728 limited by a sales order quote item BPVT 1730 which in turn is further limited by an available-to-promise BPVT 1732. The customer relationship management deployment unit 1703 further includes a sales order processing deployment unit 1730 that is limited by a sell-from-stock BPVT 1732 that in turn is further limited by an available-to-promise BPVT 1734, a pricing BPVT 1736, and a free goods processing BPVT 1738. In addition, an inbound process agent of the customer relationship management deployment unit 1704 is coupled to an outbound process agent of a purchase order processing at customer process component 1704 which is implemented at a customer site.

[0094] The logistics execution deployment unit 1705 includes a site logistics processing process component 1744 that is limited by a standard shipping BPVT 1746 that in turn is further limited by a warehouse orders BPVT 1748. The site logistics processing process component 1748 is further coupled via an outbound process agent to an accounting process component 1742 external to the logistics execution deployment unit 1705. The logistics execution deployment unit further includes an outbound delivery processing process component 1750 that is limited by a standard outbound delivery BPVT 1752 and which is coupled, via an outbound process agent, to an inbound delivery processing at customer process

component 1758 which resides at a customer site. In addition, the logistics execution deployment unit includes a confirmation and inventory process component 1754 which is limited by a confirmation and inventory posting for site logistic processes BPVT 1758.

[0095] The customer invoicing deployment unit 1708 includes a customer invoice processing process component 1758 that is limited by a standard customer invoice BPVT 1760 and which is further coupled via outbound process agents to the accounting process component 1742 and a supplier invoice processing at customer process component 1714 residing at a customer site.

[0096] The due item management deployment unit 1710 comprises a due item processing process component 1762 that is limited by a due payment order BPVT 1764 and which is coupled, via an outbound process agent, to an accounting process component 1742.

[0097] The financial accounting deployment unit 1712 includes the accounting process component which is coupled, via an outbound process agent, to another accounting process component 1766 that is defined by an accounting for sales BPVT 1768 and a profit center accounting BPVT 1770.

[0098] The payment deployment unit 1714 includes a payment processing process component 1772 that is limited by an incoming bank transfer / direct credit BPVT 1774, an incoming credit card payment PCT 1776, an incoming direct debit BPVT 1778, an incoming bill of exchange BPVT 1780, an incoming check payment BPVT 1782, and an incoming lockbox payment BPVT 1784. The payment deployment unit 1714 is also coupled to the accounting process component 1742 via an outbound process agent, a payment processing at business partner process component 1786 via inbound and outbound process agents, a payment order processing at house bank process component 1788 via an inbound process agent, and a bank statement creation at bank process component 1790 via an inbound process agent. In the payment processing process component 1772, at least one of the BPVTs 1774, 1776, 1778, 1780, 1782, and 1784 has to be used. How a customer pays (with check or credit card) is a decision not specifically determined by the integration scenario.

[0099] The subject matter described in this specification and all of the functional operations described in this specification can be implemented in digital electronic circuitry, or in computer software, firmware, or hardware, including the structural means disclosed in this specification and structural equivalents thereof, or in combinations of them. The subject matter described in this specification can be implemented as one or more computer program products, i.e., one or more computer programs tangibly embodied in an information carrier, e.g., in a machine-readable storage device or in a propagated signal, for execution by, or to control the operation of, data processing apparatus, e.g., a programmable processor, a computer, or multiple computers. A computer program (also known as a program, software, software applica-

tion, or code) can be written in any form of programming language, including compiled or interpreted languages, and it can be deployed in any form, including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A computer program does not necessarily correspond to a file. A program can be stored in a portion of a file that holds other programs or data, in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub-programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers at one site or distributed across multiple sites and interconnected by a communication network.

[0100] The processes and logic flows described in this specification can be performed by one or more programmable processors executing one or more computer programs to perform functions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application-specific integrated circuit).

[0101] Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read-only memory or a random access memory or both. The essential elements of a computer are a processor for executing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto-optical disks, or optical disks. Information carriers suitable for embodying computer program instructions and data include all forms of non-volatile memory, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

[0102] To provide for interaction with a user, the subject matter described in this specification can be implemented on a computer having a display device, e.g., a CRT (cathode ray tube) or LCD (liquid crystal display) monitor, for displaying information to the user and a keyboard and a pointing device, e.g., a mouse or a trackball, by which the user can provide input to the computer. Other kinds of devices can be used to provide for interaction with a user as well; for example, feedback provided to the user can be any form of sensory feedback, e.g., visual feedback, auditory feedback, or tactile feedback; and input from the user can be received in any form, including acoustic, speech, or tactile input.

[0103] The subject matter described in this specification can be implemented in a computing system that includes a back-end component (e.g., a data server), a middleware component (e.g., an application server), or a front-end component (e.g., a client computer having a graphical user interface or a Web browser through which a user can interact with an implementation of the subject matter described herein), or any combination of such back-end, middleware, and front-end components. The components of the system can be interconnected by any form or medium of digital data communication, e.g., a communication network. Examples of communication networks include a local area network ("LAN") and a wide area network ("WAN"), e.g., the Internet.

[0104] The computing system can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

[0105] The subject matter has been described in terms of particular variations, but other variations can be implemented and are within the scope of the following claims. For example, the actions recited in the claims can be performed in a different order and still achieve desirable results. As one example, the processes depicted in the accompanying figures do not necessarily require the particular order shown, or sequential order, to achieve desirable results. In certain implementations, multitasking and parallel processing may be advantageous. Other variations are within the scope of the following claims.

Claims

1. A computer-implemented method comprising:

defining one or more process components, each of the process components characterizing software implementing respective and distinct business processes and defining at least one process agent, each process agent enabling communications between a business object associated with the corresponding process component and a business object associated with any other process component; and
defining one or more Business Process Variant Types for at least one of the process components, each of the Business Process Variant Types associating one or more of the process agents defined for the corresponding process component, wherein selection of a process variant type causes the associated one or more process agents to be activated.

2. A method as in claim 1, wherein the process agents are outbound process agents.

3. A method as in claim 1 or 2, wherein the process components characterize inbound operations to handle incoming messages associated with a modification of reading of data encapsulated in a business object associated with the process component.

4. A method as in claim 1 or 2, wherein the process components characterize outbound operations to handle outgoing messages associated with a modification or reading of data encapsulated in at least one business object associated with another process component.

5. A method as in claim 4, wherein the outbound operations are called after the business object associated with a corresponding outbound operation is read or modified.

6. A method as in claim 5, wherein the outbound operations send messages after they are called.

7. A method as in claim 3, wherein one or more of the inbound operations is a synchronous operation operable to receive a synchronous message generated by an external synchronous outbound operation defined by an external process component.

8. A method as in claim 4, wherein one or more of the outbound operations is an asynchronous outbound operation operable to generate an asynchronous message for receipt by an asynchronous inbound operation defined by an external process component.

9. A method as in claim 3, wherein one or more of the inbound operations is operable to receive a message of a first type and convert it into a message of a second type.

10. A method according to any one of the preceding claims, wherein at least one of the process agents is an inbound process agent characterizing an inbound operation to handle incoming messages.

11. A method according to any one of the preceding claims, wherein at least one of the process agents is an outbound process agent characterizing an outbound operation to transmit outgoing messages to an external process component.

12. A method according to any one of the preceding claims, wherein a first of the process components is associated with a first deployment unit and a second of the process components is associated with a second deployment unit, the deployment units characterizing independently operable software.

13. A method according to any one of the preceding

claims, wherein the process components define service interfaces having pair-wise interactions between pairs of process components.

- 14.** A method according to any one of the preceding claims, further comprising:

defining one or more business objects, each business object being associated with a single process component.

- 15.** A method as in claim 14, wherein none of the business objects of any one of the process components interacts directly with any of the business objects associated with any of the other process components.

- 16.** A method according to any one of the preceding claims, further comprising:

logically associating a plurality of process components to realize a business scenario.

- 17.** A method according to any one of the preceding claims, further comprising:

limiting interactions between process components not having a process variant type activating process agents coupled the process components.

- 18.** A method according to any one of the preceding claims, further comprising:

verifying that each coupled process component has a corresponding process variant type activating an outbound process agent.

- 19.** A computer-implemented method of defining interactions between two process components, the method comprising:

defining, for each process component, a plurality of process agents, each process agent being either an inbound process agent or an outbound process agent, an inbound process agent being operable to receive a message from an inbound operation, an outbound process agent being operable to cause an outbound operation to send a message; and
defining interactions between at least one inbound process agent of a first process component and at least one outbound process agent of a second process component;
defining interactions between at least one inbound process agent of the second process component and at least one outbound process agent of the first process component; and

defining one or more Business Process Variant Types for at least one of the process components, each of the Business Process Variant Types associating one or more of the process agents defined for the corresponding process component, wherein selection of a process variant type causes the associated one or more process agents to be activated.

- 20.** A computer-implemented method comprising:

displaying, in a first view, a process interaction map illustrating interactions among a plurality of process components linked together by selected Business Process Variant Types, each of the process components characterizing software implementing a respective and distinct business process, and each of the process components defining a respective service interface for interacting with other process component;
displaying, in a second view, a process component architectural design illustrating an inbound part, a business object part, and an outbound part for a selected process component, the inbound part identifying all external process components that use one or more inbound operations of the selected process component, the business object part identifying all business objects associated with the selected process component, the outbound part identifying all external process components utilized by one or more outbound operations of the selected component;
and displaying, in a third view, a process component interaction architectural design illustrating message transfer between exactly two process components to modify or read business objects associated with each business object, each process component illustrating a plurality of process agents, each process agent being either an inbound process agent or an outbound process agent, the inbound process agent being operable to receive a message from an inbound operation, the outbound process agent being operable to cause an outbound operation to send a message, wherein only outbound process agents associated with a selected process variant type are activated.

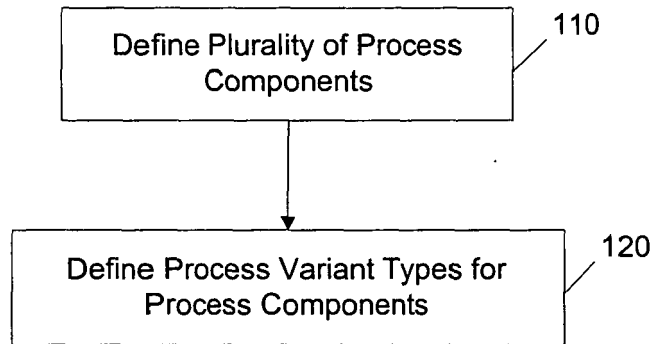


FIG. 1

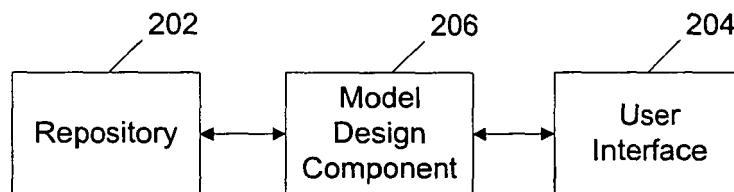


FIG. 2

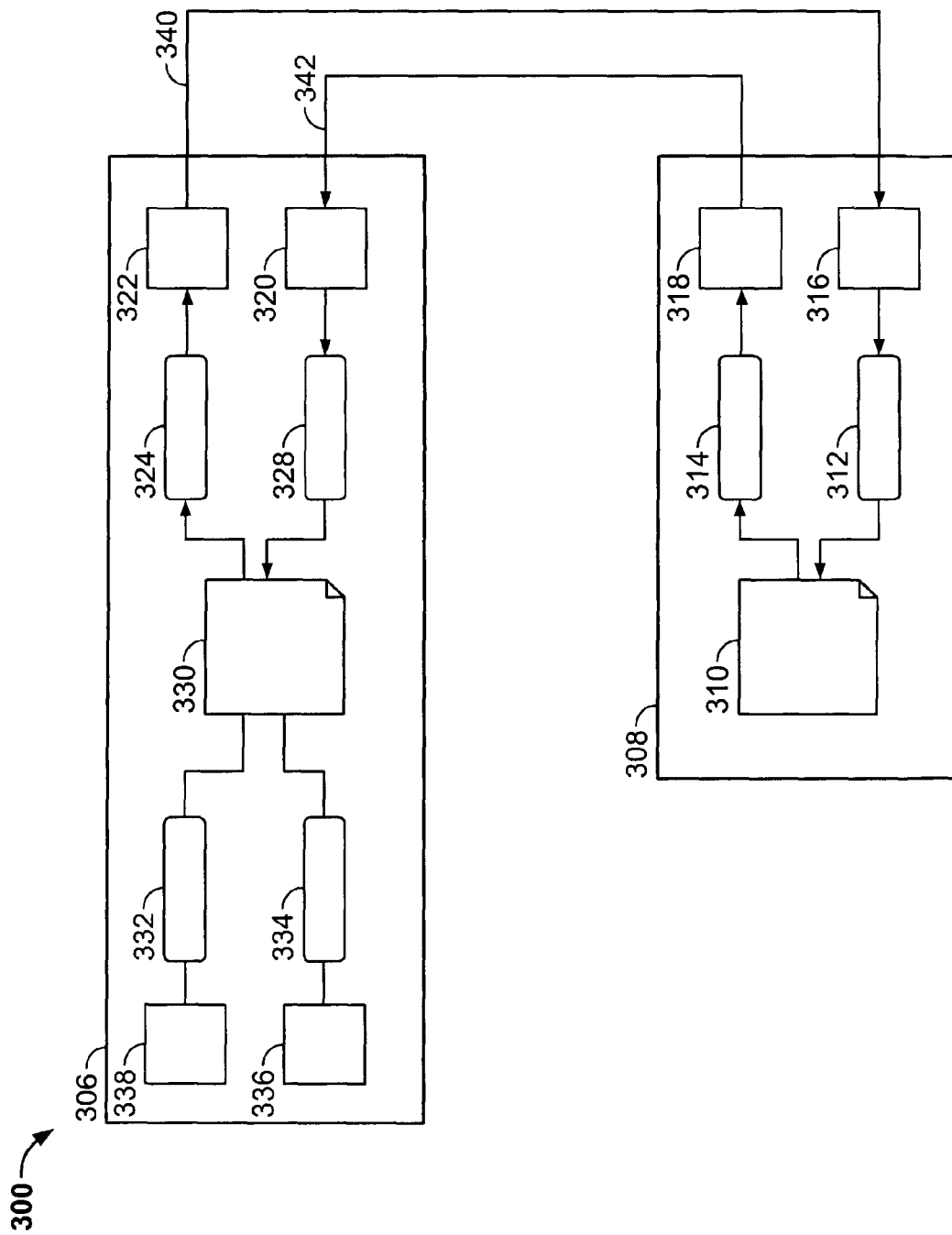


FIG. 3

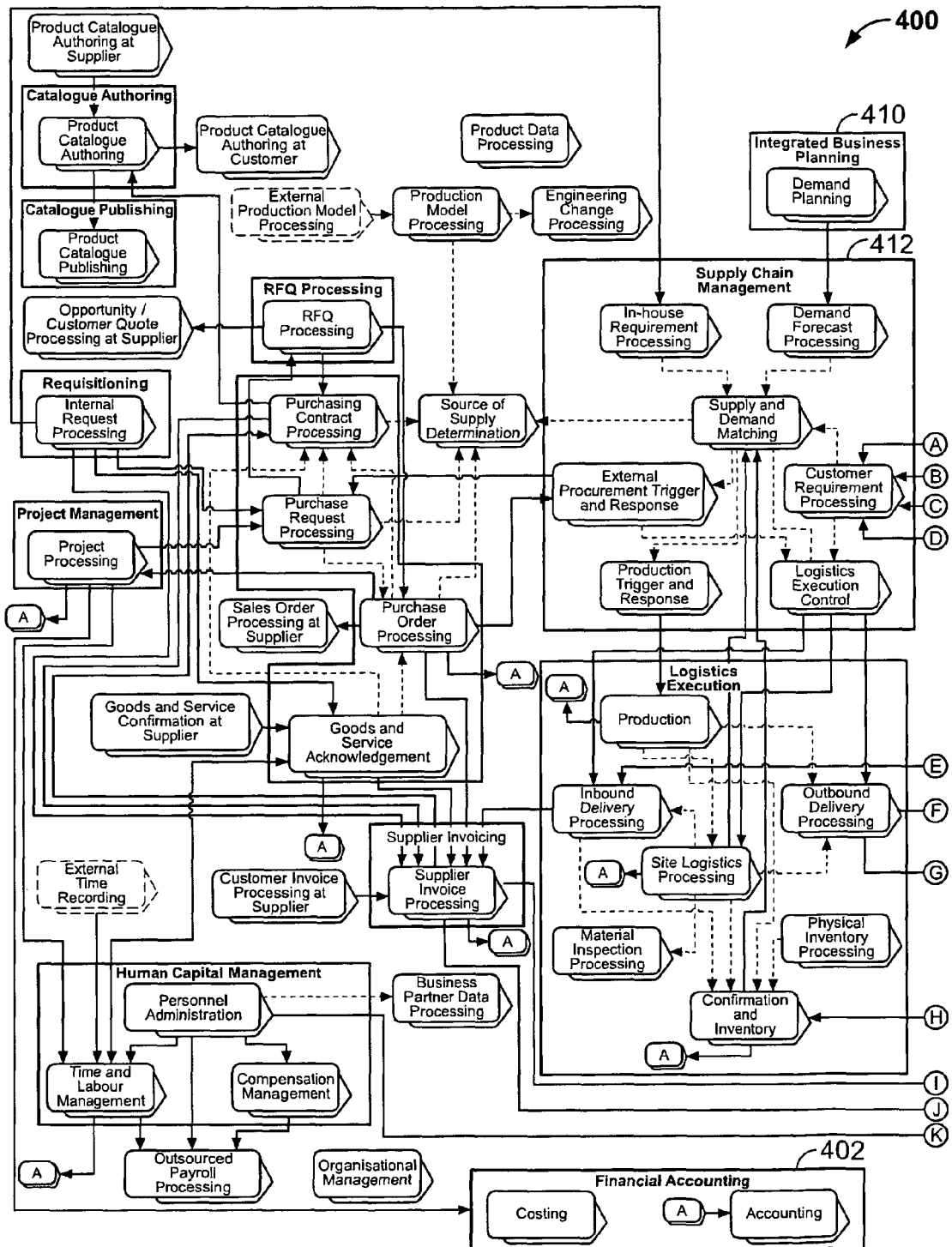
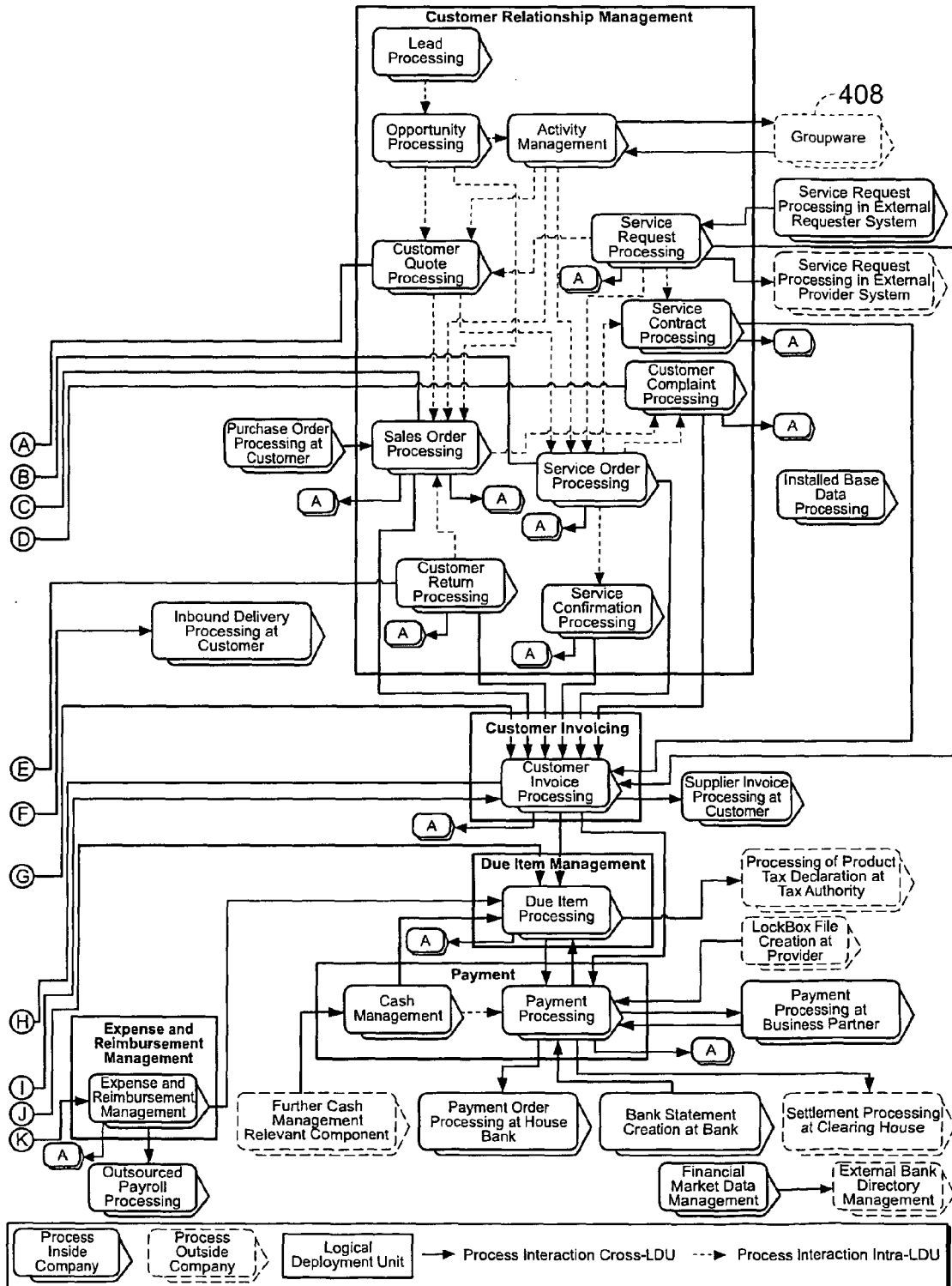


FIG. 4A



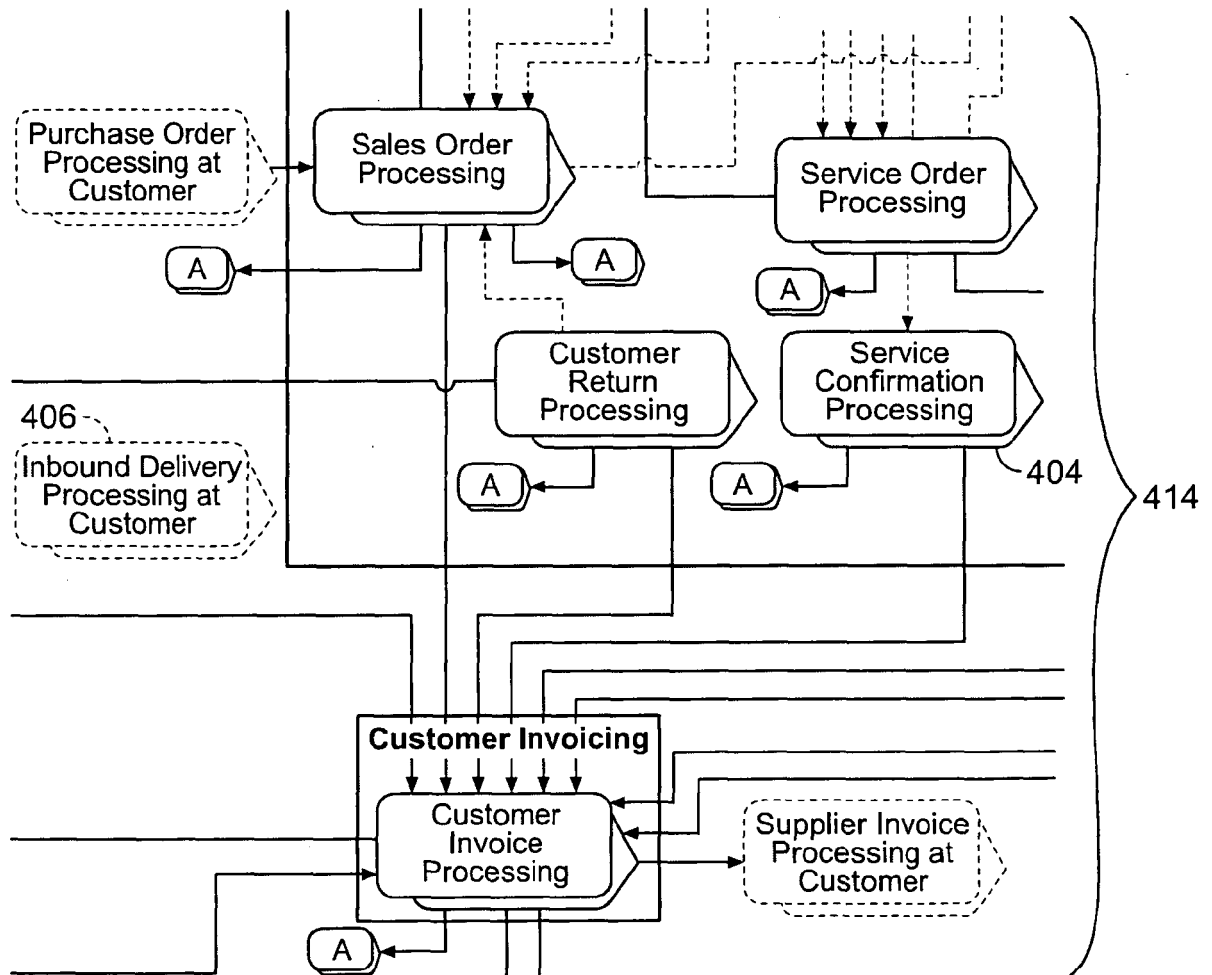


FIG. 4C

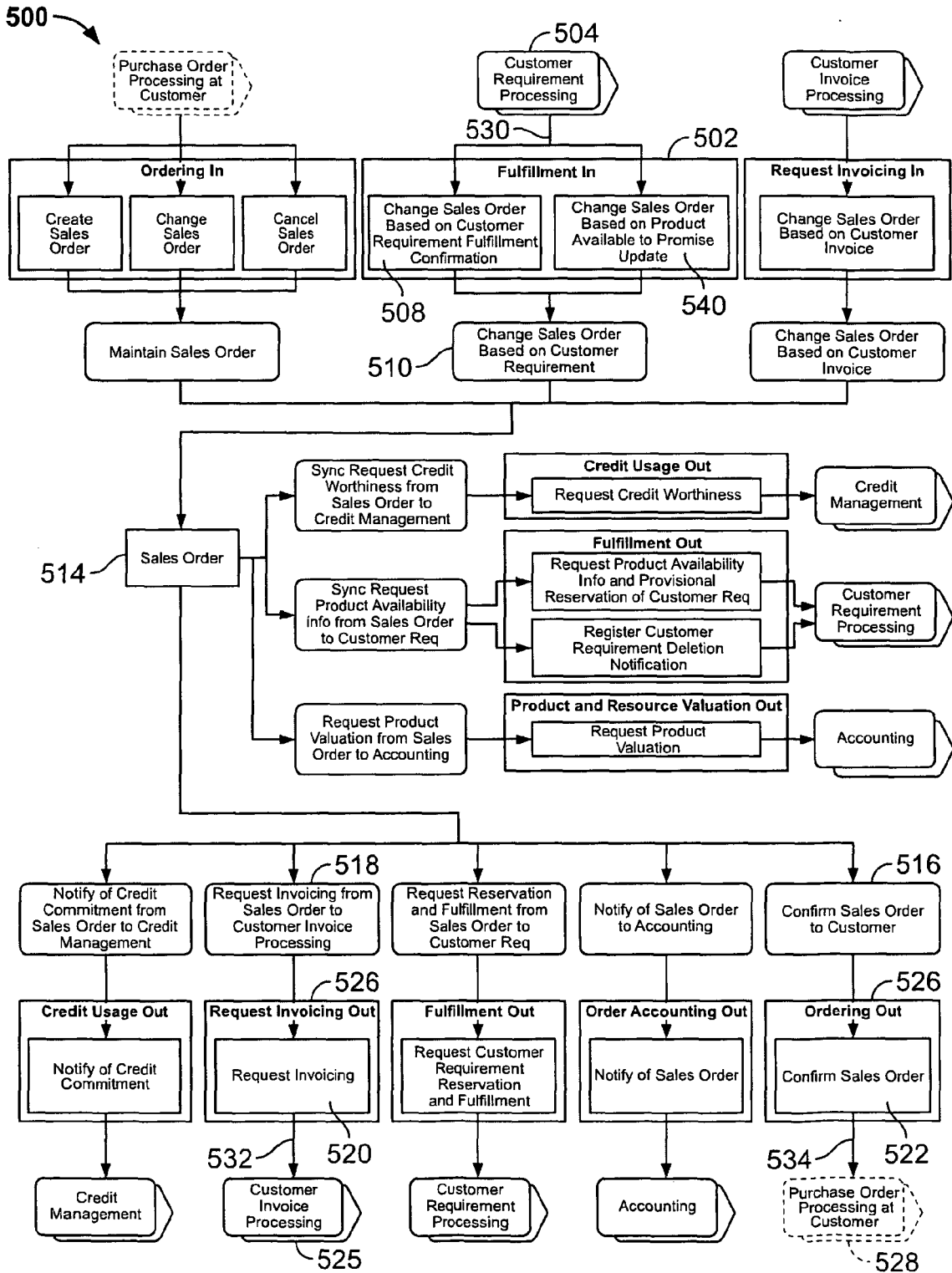


FIG. 5

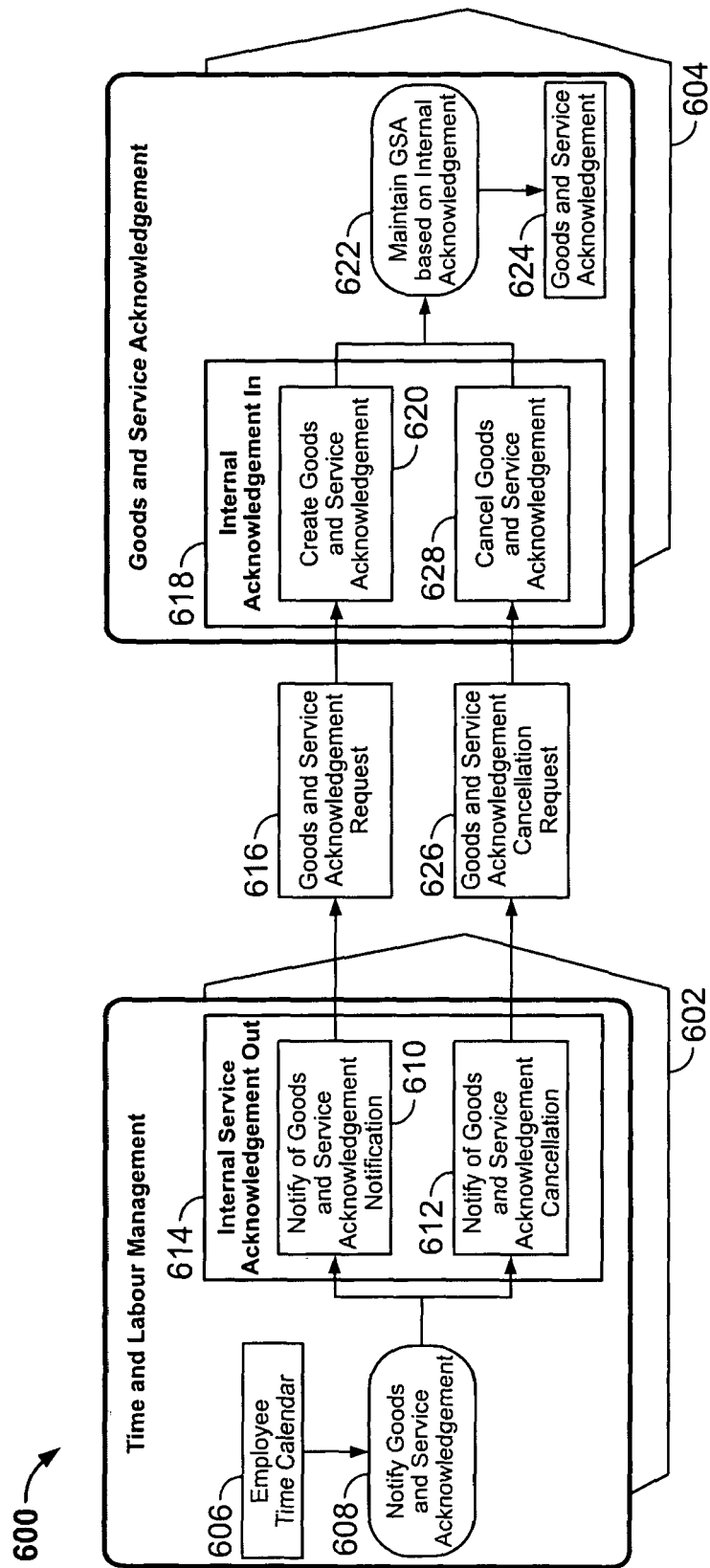


FIG. 6

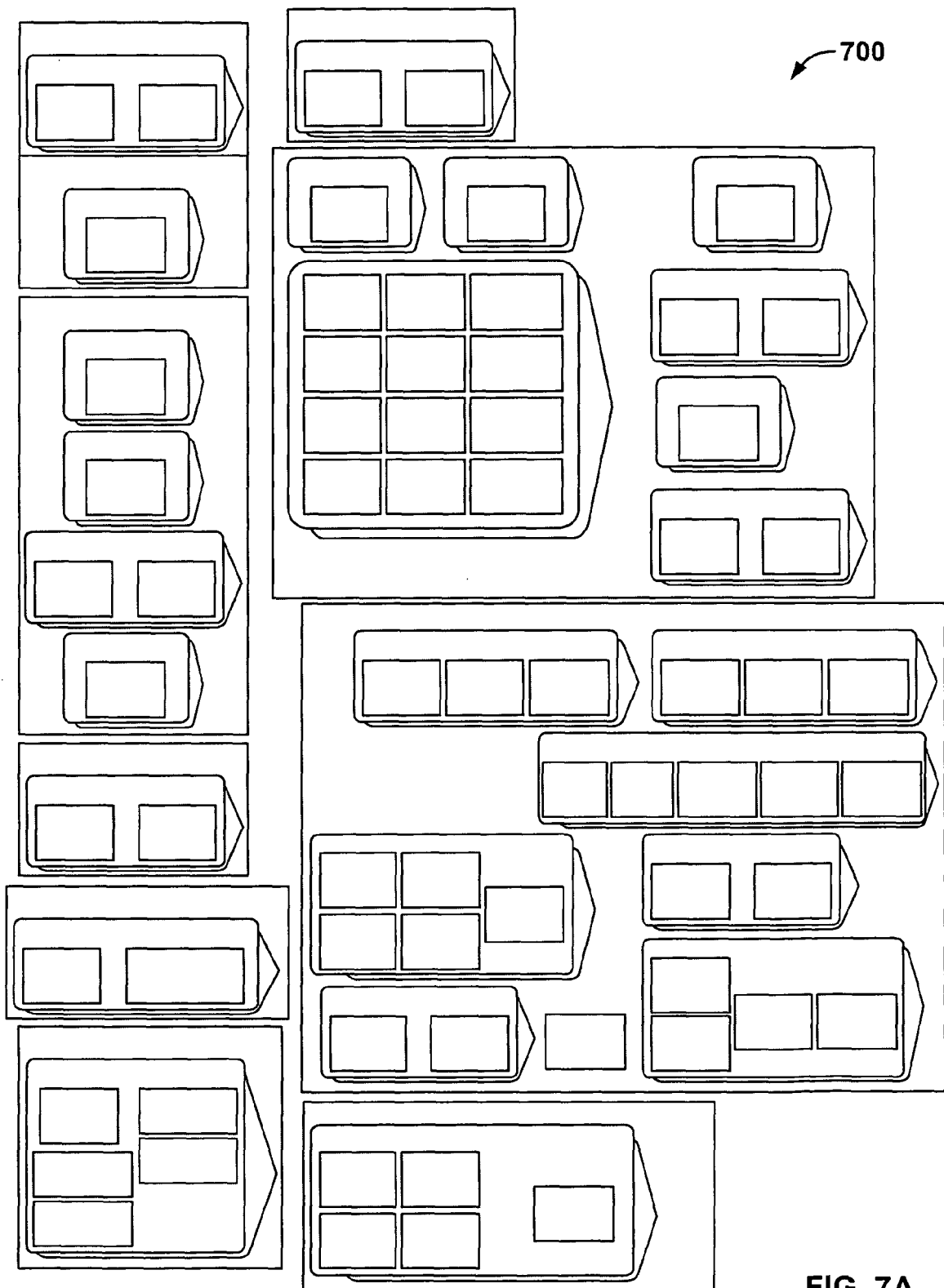


FIG. 7A

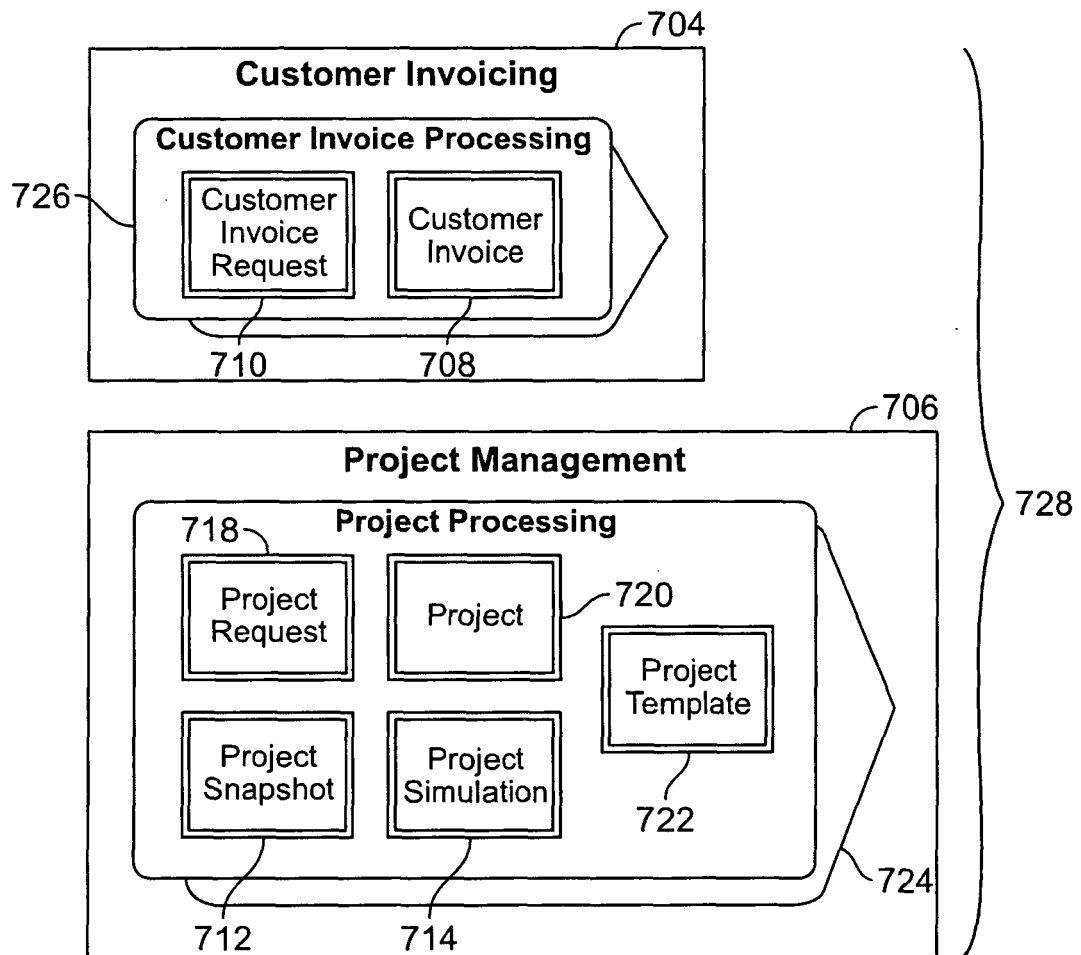


FIG. 7B

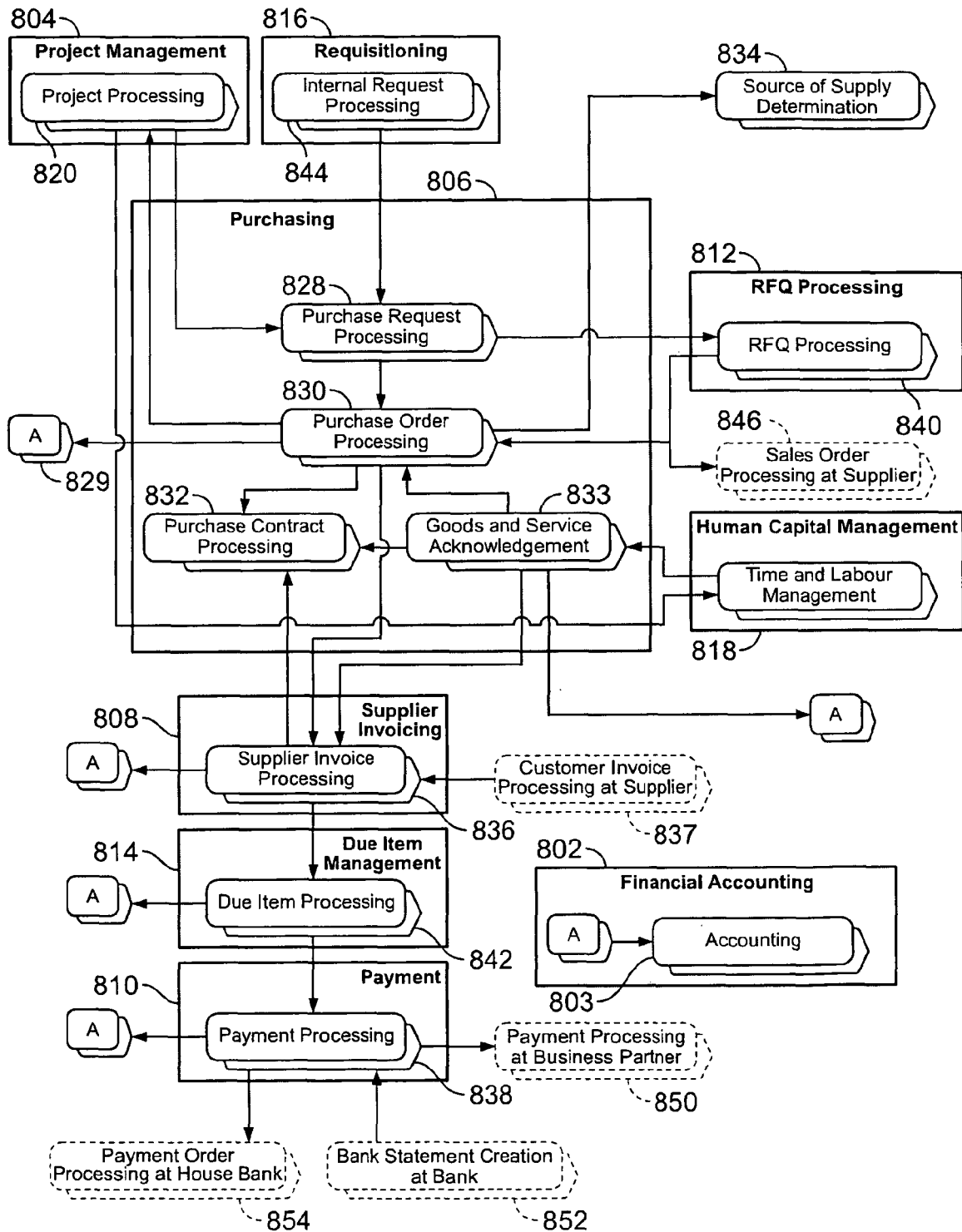


FIG. 8

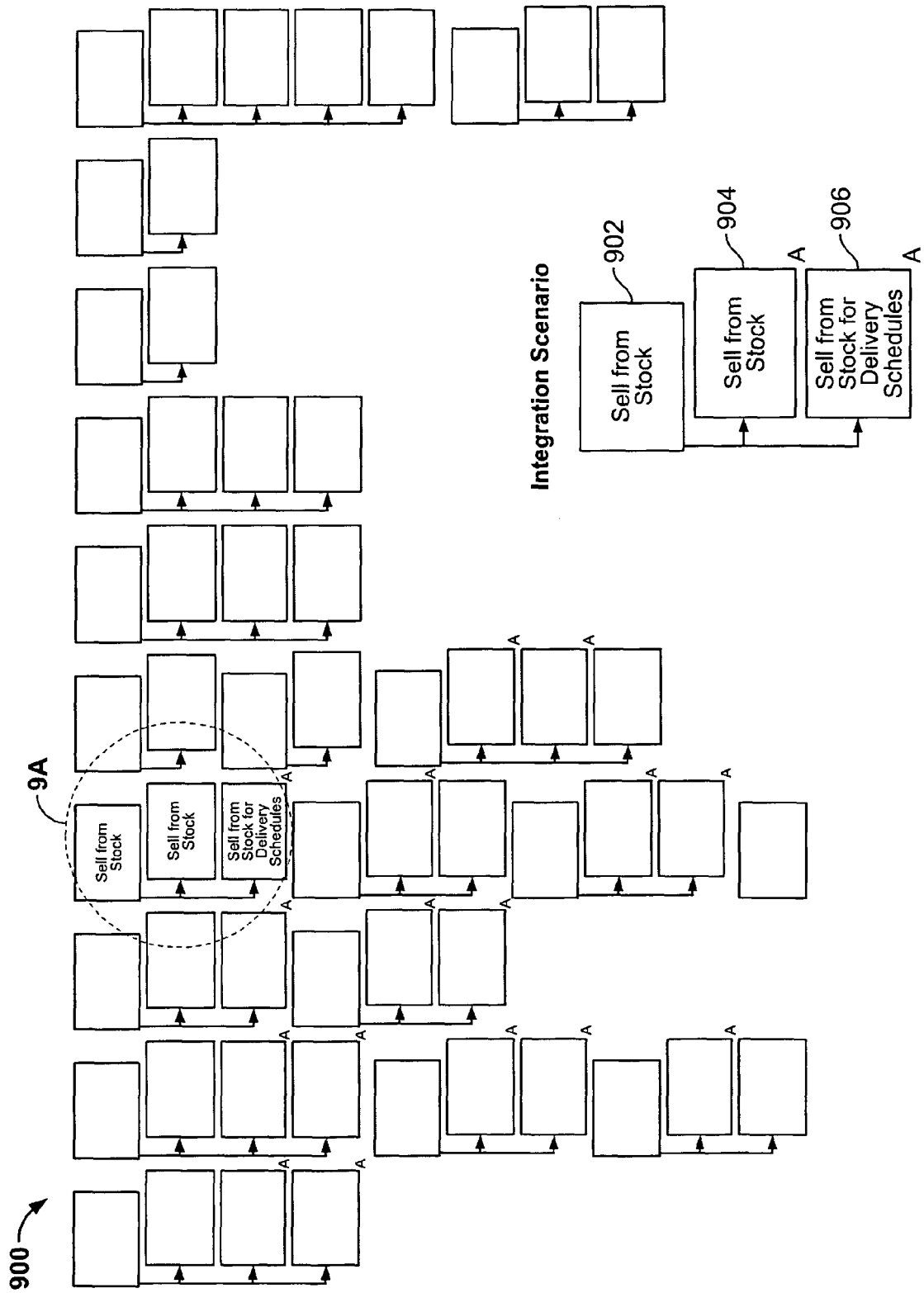


FIG. 9A

FIG. 9

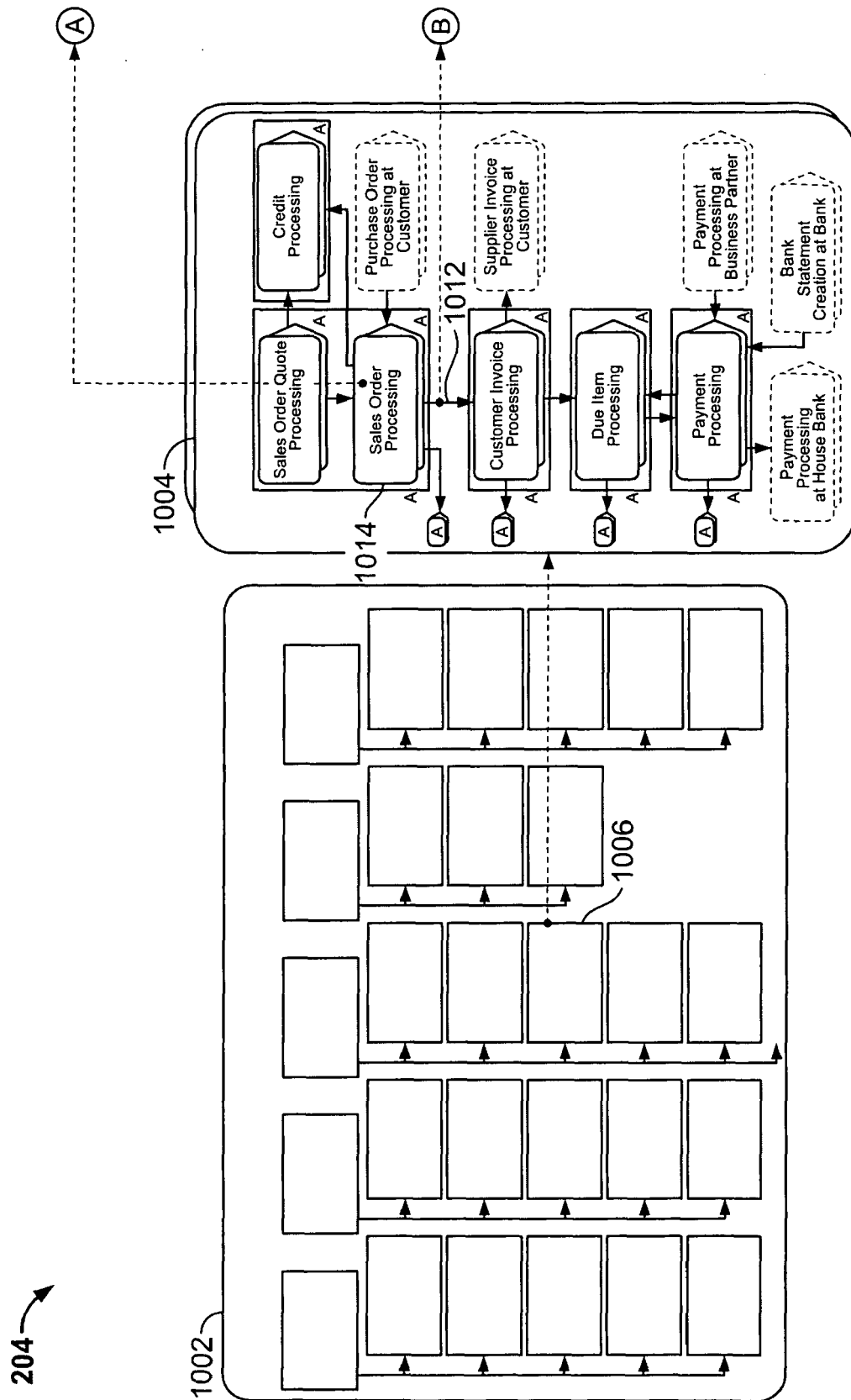


FIG. 10A

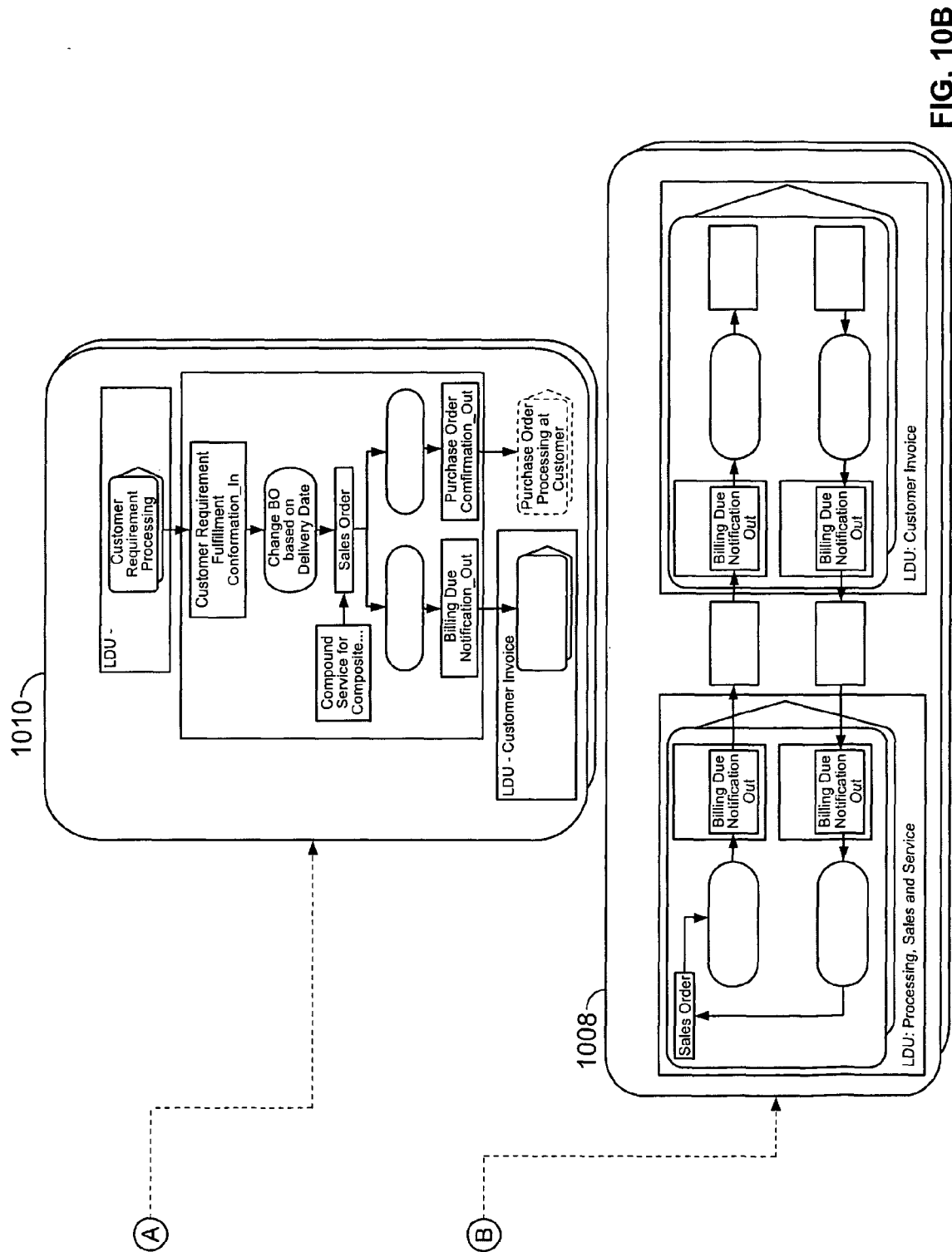


FIG. 10B

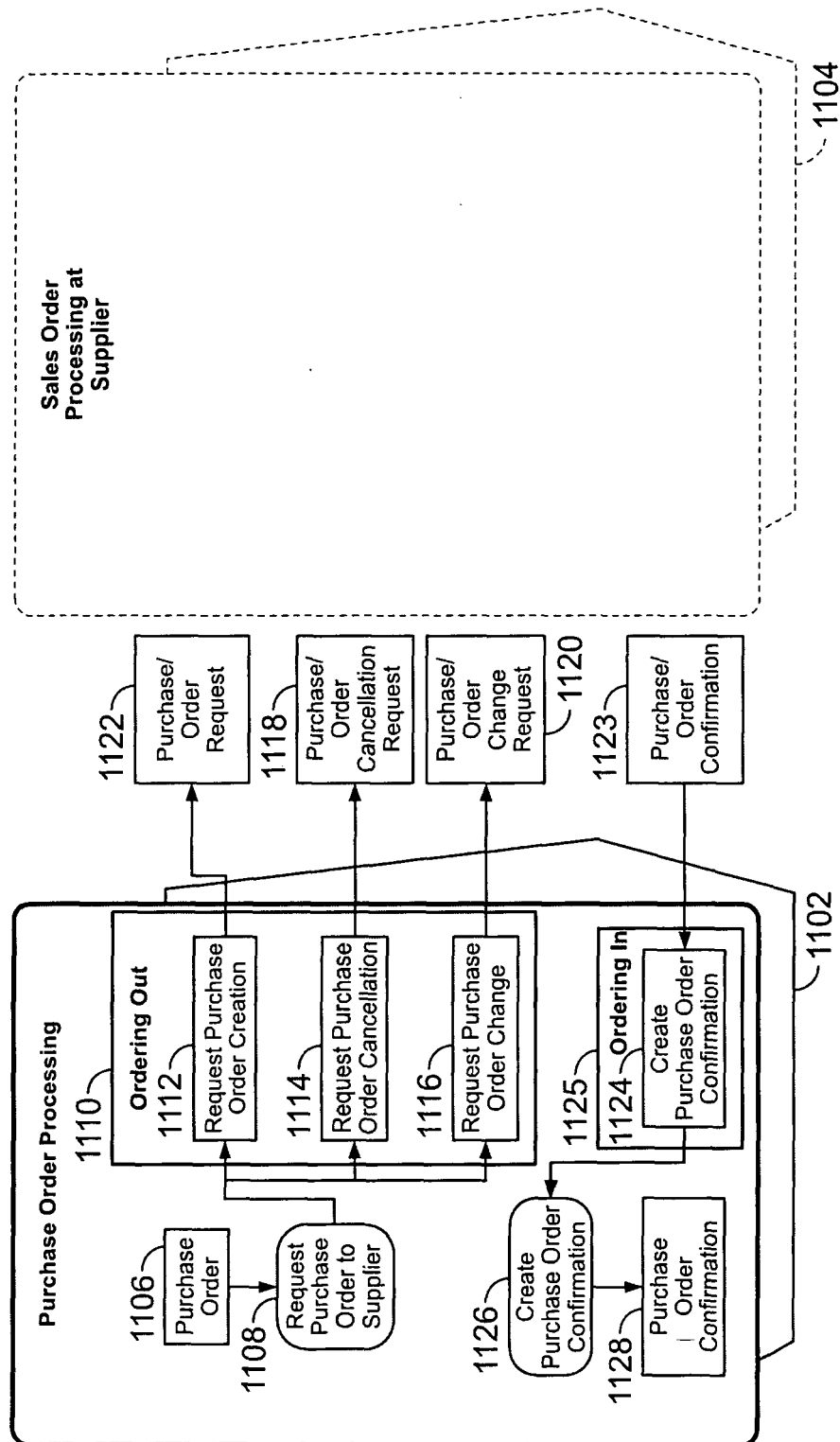


FIG. 11

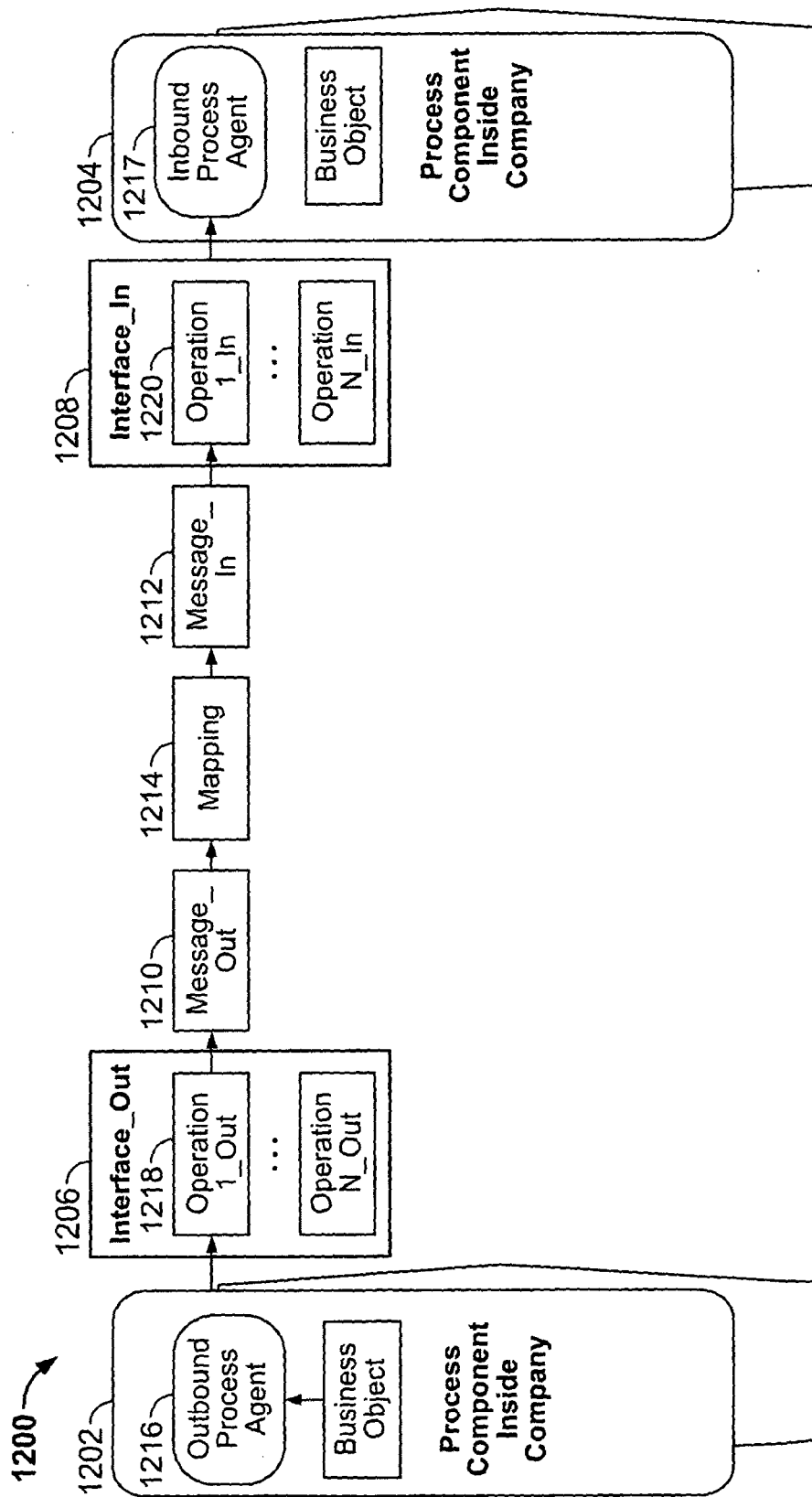


FIG. 12

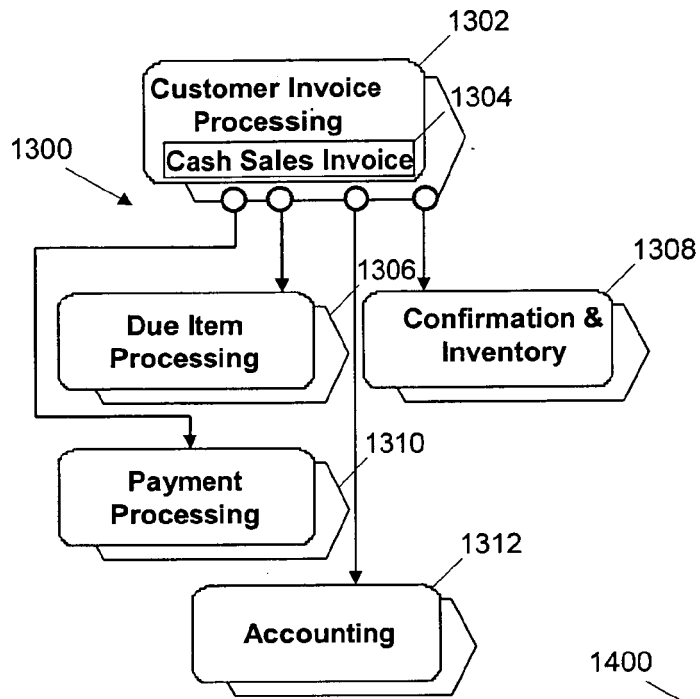


FIG. 13

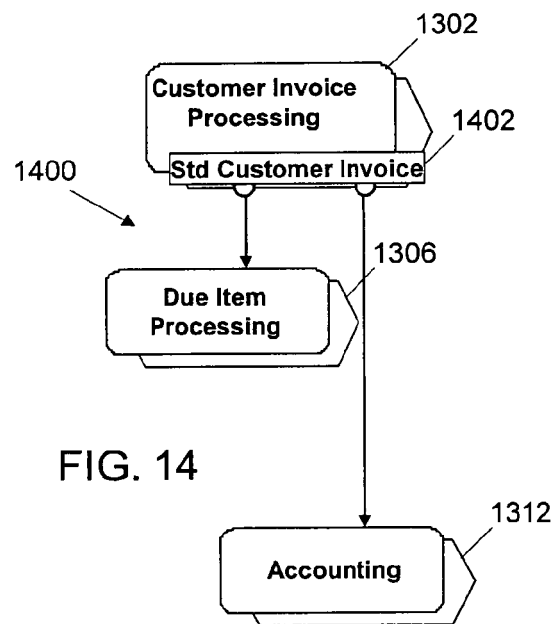


FIG. 14

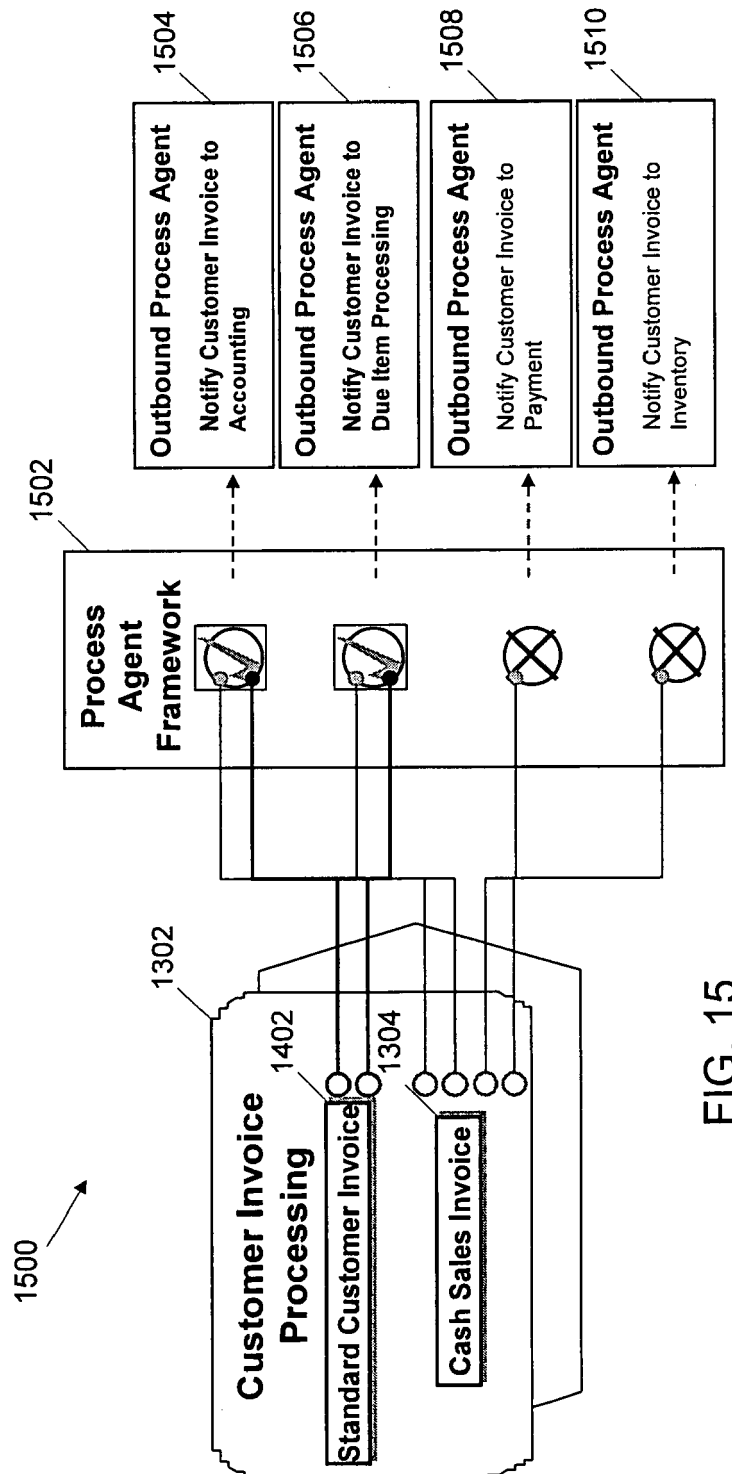
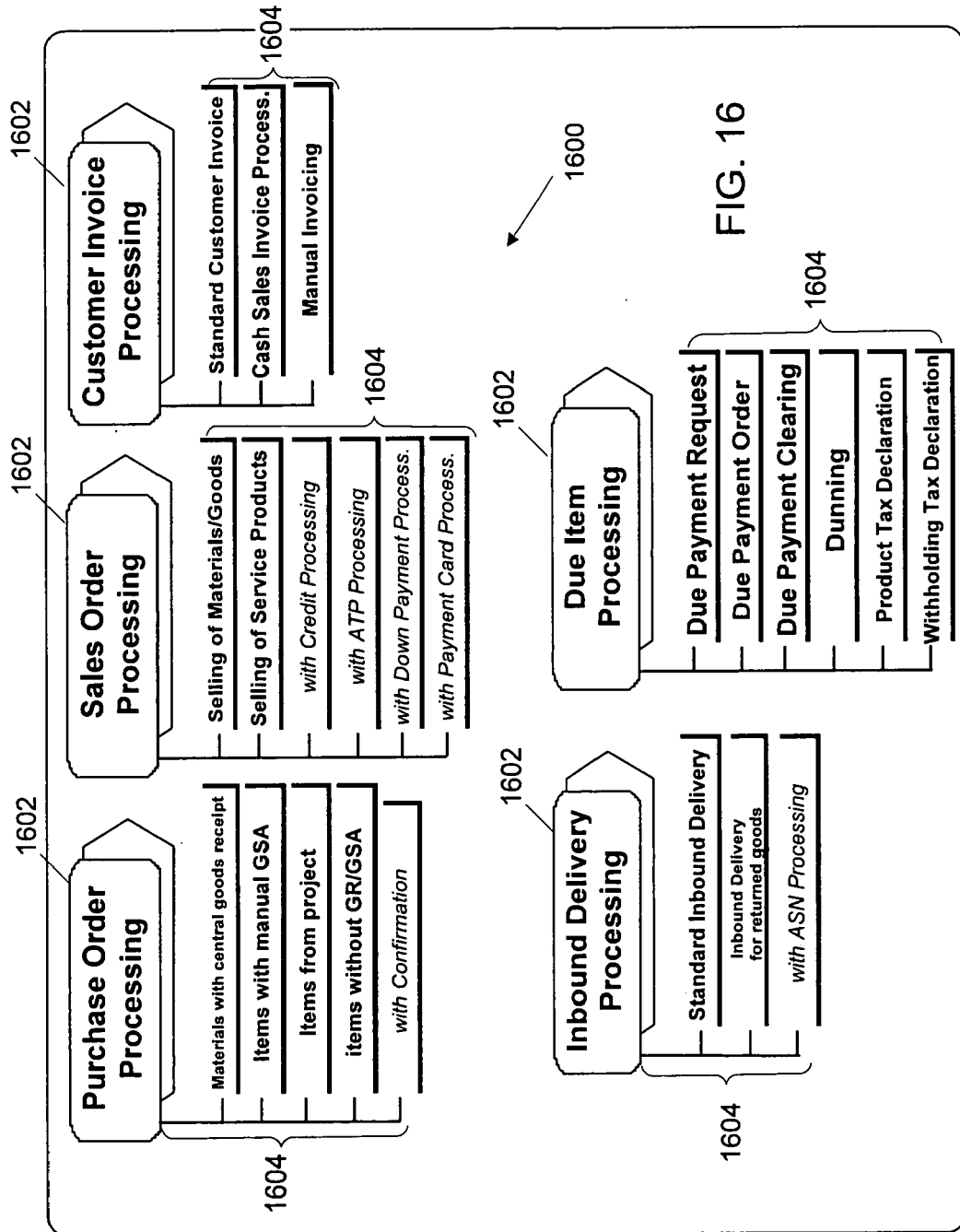


FIG. 15



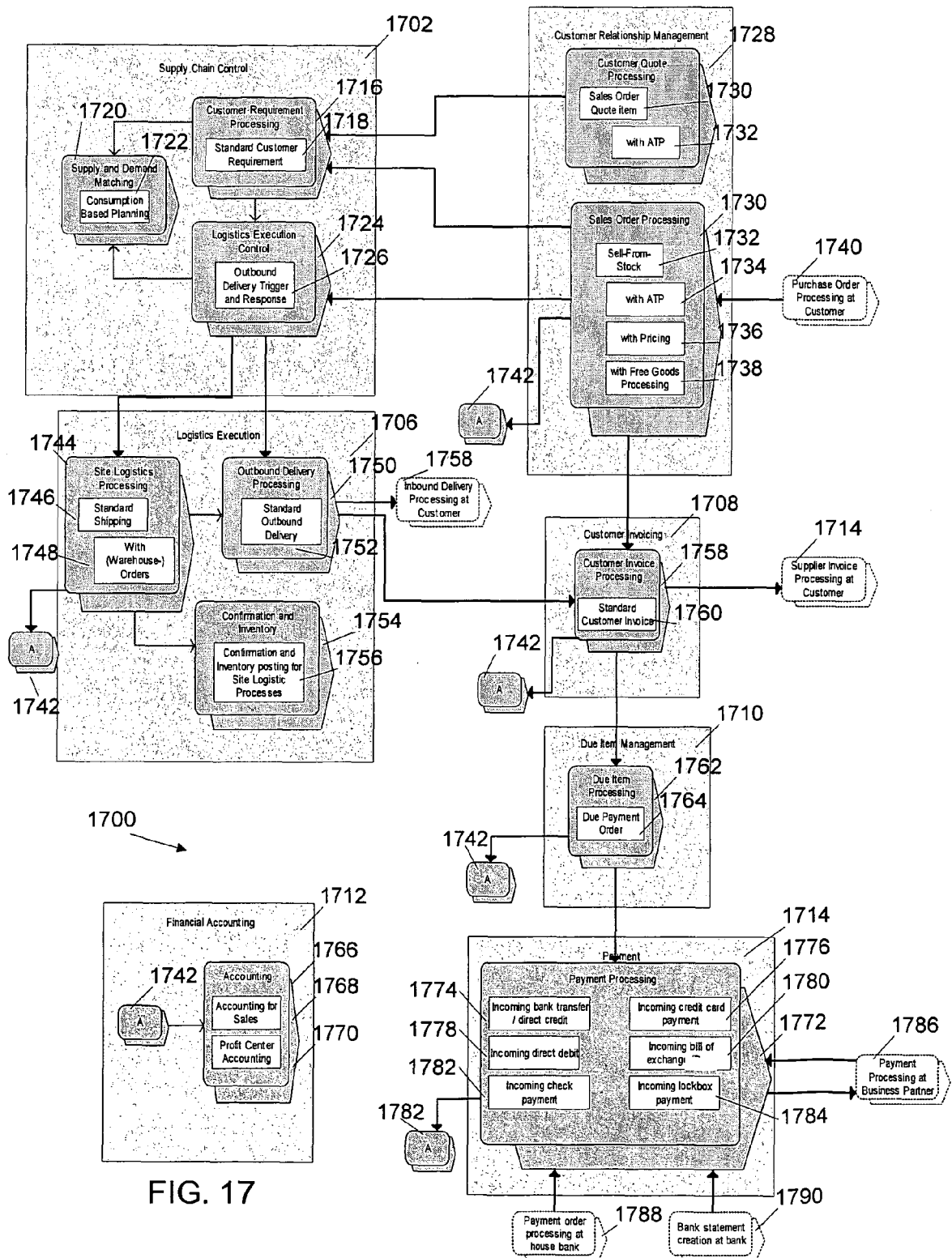


FIG. 17